Results for "((query and transfer and data and database and relationship and class)<in>metadata)"

e-mail printer friendly

Your search matched 1 of 1415139 documents.

A maximum of 100 results are displayed, 25 to a page, sorted by Relevance in Descending order.

» Search Options

View Session History

New Search

Modify Search

((query and transfer and data and database and relationship and class)<in>metadata Search

Check to search only within this results set

» Key

IEEE JNL	IEEE Journal or Magazine
IEE JNL	IEE Journal or Magazine
IEEE CNF	IEEE Conference Proceeding
IEE CNF	IEE Conference Proceeding
IEEE STD	IEEE Standard

Display Format: Citation Citation & Abstract

view selected items Select All Deselect All

1. **Logic-based query optimization for object databases**
Grant, J.; Gryz, J.; Minker, J.; Raschid, L.;
Knowledge and Data Engineering, IEEE Transactions on
Volume 12, Issue 4, July-Aug. 2000 Page(s):529 - 547
Digital Object Identifier 10.1109/69.868906
AbstractPlus | References | Full Text: PDF(328 KB) IEEE JNL
Rights and Permissions

Indexed by Inspec®

© Copyright 2006 IEEE – All Rights Reserved

Results for "((query and transfer and data and database and relationship)<in>metadata)"

e-mail printer friendly

Your search matched 3 of 1415139 documents.

A maximum of 100 results are displayed, 25 to a page, sorted by Relevance in Descending order.

» Search Options

View Session History

New Search

Modify Search

((query and transfer and data and database and relationship)<in>metadata) Search

Check to search only within this results set

» Key

Display Format: Citation Citation & Abstract

IEEE JNL	IEEE Journal or Magazine
IEE JNL	IEE Journal or Magazine
IEEE CNF	IEEE Conference Proceeding
IEE CNF	IEE Conference Proceeding
IEEE STD	IEEE Standard

view selected items Select All Deselect All

1. **Logic-based query optimization for object databases**
Grant, J.; Gryz, J.; Minker, J.; Raschid, L.;
Knowledge and Data Engineering, IEEE Transactions on
Volume 12, Issue 4, July-Aug. 2000 Page(s):529 - 547
Digital Object Identifier 10.1109/69.868906
AbstractPlus | References | Full Text: PDF(328 KB) IEEE JNL
Rights and Permissions

2. **Determining the optimal file size on tertiary storage systems based on the distribution of query sizes**
Bernardo, L.M.; Nordberg, H.; Rotem, D.; Shoshani, A.;
Scientific and Statistical Database Management, 1998. Proceedings. Tenth International Conference on
1-3 July 1998 Page(s):22 - 31
Digital Object Identifier 10.1109/SSDM.1998.688108
AbstractPlus | Full Text: PDF(144 KB) IEEE CNF
Rights and Permissions

3. **Optimization Models for Configuring Distributed Computer Systems**
Gavish, B.;
Computers, IEEE Transactions on
Volume C-36, Issue 7, Jul 1987 Page(s):773 - 793
AbstractPlus | Full Text: PDF(5184 KB) IEEE JNL
Rights and Permissions

© Copyright 2006 IEEE – All Rights Reserved

Indexed by Inspec

IEEE Xplore® RELEASE 2.1

Home | Login | Logout | Access Information | Alerts | Sitemap | Help

Welcome United States Patent and Trademark Office

AbstractPlus BROWSE SEARCH IEEE XPLORE GUIDE SUPPORT

View Search Results

printer friendly

Access this document

Full Text: PDF (536 KB)

Architecture and Performance of a Grid-Enabled Lookup-Based Biomedical Optimization Application: Light Scattering Spectroscopy

Figueiredo J. Backman Vadim Liu Yang Paladugula Jithendar

This paper appears in: IEEE Transactions on Information Technology in Biomedicine : Accepted for future publication
Digital Object Identifier: 10.1109/TITB.2006.876032
ISSN: 1089-7771

Abstract

This paper presents a case study of a Grid-enabled implementation of Light Scattering Spectroscopy (LSS). The LSS technique allows non-invasive detection of pre-cancerous changes in human epithelium, differentiating from traditional biopsies by allowing in-vivo diagnosis of tissue samples and quantitative analyses of parameters related to cancerous changes via numerical techniques. The paper describes the architecture of GridLSS and its integration with a Web-based Grid computing portal. GridLSS solves an optimization problem of determining the light scattering spectrum that best fits experimental spectral data among a large set of spectra computed analytically using rigorous Mie theory. The novel approach taken in the paper is based on the pre-computation and storage of Mie theory spectra in look-up **databases** that are **queried** during the minimization process. The paper makes three important contributions: 1) it presents a novel parallel **application** for LSS analysis that delivers high performance in wide-area distributed computing environment; 2) it evaluates and analyzes the performance of this **application** in cluster-based high performance computing environments that are typical of Grid deployments; and 3) it shows that the performance of GridLSS benefits significantly from the use of on-demand Grid **data transfers** based on virtualized distributed file systems and from user-level caches for remote file system data.

Index Terms

biomedical applications Grid computing spectroscopy virtualization

View Search Results

Help Contact Us Privacy & Security IEEE.org

Indexed by Inspec®

© Copyright 2006 IEEE – All Rights Reserved

Results for "((query and (transfer<near/2>data) and interaction and database)<in>metadata)"

e-mail printer friendly

Your search matched 0 documents.

A maximum of 100 results are displayed, 25 to a page, sorted by Relevance in Descending order.

» Search Options

View Session History

New Search

Modify Search

((query and (transfer<near/2>data) and interaction and database)<in>metadata) Search

Check to search only within this results set

Display Format: Citation Citation & Abstract

» Key

IEEE JNL	IEEE Journal or Magazine
IEE JNL	IEE Journal or Magazine
IEEE CNF	IEEE Conference Proceeding
IEE CNF	IEE Conference Proceeding
IEEE STD	IEEE Standard

No results were found.

Please edit your search criteria and try again. Refer to the Help pages if you need assistance revising your search.

© Copyright 2006 IEEE – All Rights Reserved

Indexed by Inspec®

ACM PORTAL
USPTO
Subscribe (Full Service) Register (Limited Service, Free) Login
Search: The ACM Digital Library The Guide
query and relationship and class and database and software ap
SEARCH
THE ACM DIGITAL LIBRARY
Feedback Report a problem Satisfaction survey

Terms used
query and **relationship** and **class** and **database** and **software application** and **data** and **transfer** and **XML**

Found **91,088** of **185,942**

Sort results by relevance
Display results expanded form
Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 21 - 40 of 200 Result page: previous 1 **2** 3 4 5 6 7 8 9 10 next
Best 200 shown
Relevance scale

21 Semantics, ontologies & enterprise integration track: Efficient integration of web services with distributed data flow and active mediation
David Liu, Jun Peng, Kincho H. Law, Gio Wiederhold
March 2004 **Proceedings of the 6th international conference on Electronic commerce ICEC '04**
Publisher: ACM Press
Full text available: pdf(254.77 KB) Additional Information: full citation, abstract, references, index terms

This paper presents a loosely coupled service-composition paradigm. This paradigm employs a distributed data flow that differs markedly from centralized information flow adopted by current service integration frameworks, such as CORBA, J2EE and SOAP. Distributed data flows support direct data transmission to avoid many performance bottlenecks of centralized processing. In addition, active mediation is used in applications employing multiple web services that are not fully compatible in terms of ...

Keywords: active mediation, direct data transmission, handheld services, mobile class, service integration, web services

22 A fine-grained access control system for XML documents
Ernesto Damiani, Sabrina De Capitani di Vimercati, Stefano Paraboschi, Pierangela Samarati
May 2002 **ACM Transactions on Information and System Security (TISSEC)**, Volume 5 Issue 2
Publisher: ACM Press
Full text available: pdf(330.60 KB) Additional Information: full citation, abstract, references, citings, index terms

Web-based applications greatly increase information availability and ease of access, which is optimal for public information. The distribution and sharing of information via the Web that must be accessed in a selective way, such as electronic commerce transactions, require the definition and enforcement of security controls, ensuring that information will be accessible only to authorized entities. Different approaches have been proposed that address the problem of protecting information in a Web ...

Keywords: Access control, World Wide Web, XML documents, authorizations specification and enforcement

23 A survey and analysis of Electronic Healthcare Record standards
Marco Eichelberg, Thomas Aden, Jörg Riesmeier, Asuman Dogac, Gokce B. Laleci
December 2005 **ACM Computing Surveys (CSUR)**, Volume 37 Issue 4
Publisher: ACM Press

Full text available: pdf(844.11 KB) Additional Information: full citation, abstract, references, index terms

Medical information systems today store clinical information about patients in all kinds of proprietary formats. To address the resulting interoperability problems, several Electronic Healthcare Record standards that structure the clinical content for the purpose of exchange are currently under development. In this article, we present a survey of the most relevant Electronic Healthcare Record standards, examine the level of interoperability they provide, and assess their functionality in terms o ...

Keywords: Electronic Healthcare Record standards, eHealth, interoperability

24 The Web Service Discovery Architecture

Wolfgang Hoschek

November 2002 **Proceedings of the 2002 ACM/IEEE conference on Supercomputing**

Publisher: IEEE Computer Society Press

Full text available: pdf(282.28 KB) Additional Information: full citation, abstract, references, index terms

In this paper, we propose the Web Service Discovery Architecture (WSDA). At runtime, Grid applications can use this architecture to discover and adapt to remote services. WSDA promotes an interoperable web service discovery layer by defining appropriate services, interfaces, operations and protocol bindings, based on industry standards. It is unified because it subsumes an array of disparate concepts, interfaces and protocols under a single semi-transparent umbrella. It is modular because it def ...

25 Supporting industrial hyperwebs: lessons in scalability

Kenneth M. Anderson

May 1999 **Proceedings of the 21st international conference on Software engineering**

Publisher: IEEE Computer Society Press

Full text available: pdf(1.47 MB) Additional Information: full citation, references, citings, index terms

Keywords: open hypermedia systems, scalability, software engineering

26 Workshop on compositional software architectures: workshop report

May 1998 **ACM SIGSOFT Software Engineering Notes**, Volume 23 Issue 3

Publisher: ACM Press

Full text available: pdf(2.91 MB) Additional Information: full citation, index terms

27 Document management: Accommodating paper in document databases

Majed AbuSafiya, Subhasish Mazumdar

October 2004 **Proceedings of the 2004 ACM symposium on Document engineering**

Publisher: ACM Press

Full text available: pdf(234.77 KB) Additional Information: full citation, abstract, references, index terms

Although the paperless office has been imminent for decades, documents in paper form continue to be used extensively in almost all organizations. Present-day information systems are designed on the premise that any paper document in use will be either converted into electronic form or merely printed from electronic file(s) accessible to the system. Yet, paper is the medium of choice in many situations, mainly owing to its portability and usability, and the medium of necessity in others, espec ...

Keywords: RFID, document databases, document management, enterprise document model, paper documents, paper manifestation

28 Paper session DB-2 (databases): security and privacy: A function-based access control model for XML databases

Naizhen Qi, Michiharu Kudo, Jussi Myllymaki, Hamid Pirahesh

October 2005 **Proceedings of the 14th ACM international conference on Information and knowledge management CIKM '05**

Publisher: ACM Press

Full text available: pdf(129.07 KB) Additional Information: full citation, abstract, references, index terms

XML documents are frequently used in applications such as business transactions and medical records involving sensitive information. Typically, parts of documents should be visible to users depending on their roles. For instance, an insurance agent may see the billing information part of a medical document but not the details of the patient's medical history. Access control on the basis of data location or value in an XML document is therefore essential. In practice, the number of access control ...

Keywords: XML database, access control, expressiveness, fine-grained access control, rule functions

29 A Web Odyssey: from Codd to XML

Victor Vianu

May 2001 **Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems**

Publisher: ACM Press

Full text available: pdf(282.10 KB) Additional Information: full citation, references, citings, index terms

30 Efficient evaluation of XML middle-ware queries

Mary Fernandez, Atsuyuki Morishima, Dan Suciu

May 2001 **ACM SIGMOD Record , Proceedings of the 2001 ACM SIGMOD international conference on Management of data SIGMOD '01**, Volume 30 Issue 2

Publisher: ACM Press

Full text available: pdf(414.15 KB) Additional Information: full citation, abstract, references, citings, index terms

We address the problem of efficiently constructing materialized XML views of relational databases. In our setting, the XML view is specified by a query in the declarative query language of a middle-ware system, called SilkRoute. The middle-ware system evaluates a query by sending one or more SQL queries to the target relational database, integrating the resulting tuple streams, and adding the XML tags. We focus on how to best choose the SQL queries, without having control over the target RDBM ...

31 Technical papers: software presentation: Towards large-scale information integration

Kenneth M. Anderson, Susanne A. Sherba, William V. Lepthien

May 2002 **Proceedings of the 24th International Conference on Software Engineering**

Publisher: ACM Press

Full text available: pdf(1.46 MB) Additional Information: full citation, abstract, references, citings, index terms

Software engineers confront many challenges during software development. One challenge is managing the relationships that exist between software artifacts. We refer to this task as information integration, since establishing a relationship between documents typically implies that an engineer must integrate information from each of the documents to perform a development task. In the past, we have applied open hypermedia techniques and technology to address this challenge. We now extend this work ...

32 Business-to-business interactions: issues and enabling technologies

B. Medjahed, B. Benatallah, A. Bouguettaya, A. H. H. Ngu, A. K. Elmagarmid

May 2003 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 12 Issue 1

Publisher: Springer-Verlag New York, Inc.

Full text available: pdf(558.34 KB) Additional Information: full citation, abstract, citings, index terms

Business-to-Business (B2B) technologies pre-date the Web. They have existed for at least as long as the Internet. B2B applications were among the first to take advantage of advances in computer networking. The Electronic Data Interchange (EDI) business standard is an illustration of such an early adoption of the advances in computer networking. The ubiquity and the affordability of the Web has made it possible for the masses of businesses to automate their B2B interactions. However, several issu ...

Keywords: B2B Interactions, Components, E-commerce, EDI, Web services, Workflows, XML

33 Supporting application development in the semantic web

Daniel Oberle, Steffen Staab, Rudi Studer, Raphael Volz
May 2005 **ACM Transactions on Internet Technology (TOIT)**, Volume 5 Issue 2
Publisher: ACM Press
Full text available: pdf(1.89 MB) Additional Information: full citation, abstract, references, index terms

The Semantic Web augments the current WWW by giving information a well-defined meaning, better enabling computers and people to work in cooperation. This is done by adding machine understandable content to Web resources. Such added content is called metadata, whose semantics is provided by referring to an ontology---a domain's conceptualization agreed upon by a community. The Semantic Web relies on the complex interaction of several technologies involving ontologies. Therefore, sophisticated Sem ...

Keywords: Application server, KAON, KAON SERVER, Semantic Web, Wonder-Web, extensibility, interoperation, middleware, ontology, reuse, semantic middleware

34 Frontmatter (TOC, Letters, Election results, Software Reliability Resources!, Computing Curricula 2004 and the Software Engineering Volume SE2004, Software Reuse Research, ICSE 2005 Forward)

July 2005 **ACM SIGSOFT Software Engineering Notes**, Volume 30 Issue 4
Publisher: ACM Press
Full text available: pdf(6.19 MB) Additional Information: full citation, index terms

35 Exchanging intensional XML data

Tova Milo, Serge Abiteboul, Bernd Amann, Omar Benjelloun, Fred Dang Ngoc
March 2005 **ACM Transactions on Database Systems (TODS)**, Volume 30 Issue 1
Publisher: ACM Press
Full text available: pdf(1.07 MB) Additional Information: full citation, abstract, references, index terms

XML is becoming the universal format for data exchange between applications. Recently, the emergence of Web services as standard means of publishing and accessing data on the Web introduced a new class of XML documents, which we call *intensional* documents. These are XML documents where some of the data is given explicitly while other parts are defined only intensionally by means of embedded calls to Web services.When such documents are exchanged between applications, one has the choice of ...

Keywords: Data exchange, Web services, XML, intensional information, typing

36 Software engineering tools and environments: a roadmap

Harold Ossher, William Harrison, Peri Tarr
May 2000 **Proceedings of the Conference on The Future of Software Engineering**
Publisher: ACM Press
Full text available: pdf(1.86 MB) Additional Information: full citation, references, citings, index terms

Keywords: integration, process-centered software engineering environments, programming support environments, separation of concerns, software engineering environments, tools

37 XQBE (XQuery By Example): A visual interface to the standard XML query language

Daniele Braga, Alessandro Campi, Stefano Ceri
June 2005 **ACM Transactions on Database Systems (TODS)**, Volume 30 Issue 2
Publisher: ACM Press
Full text available: pdf(2.93 MB) Additional Information: full citation, abstract, references, index terms

The spreading of XML data in many contexts of modern computing infrastructures and systems causes a pressing need for adequate XML querying capabilities; to address this need, the W3C is proposing XQuery as the standard query language for XML, with a language paradigm and a syntactic flavor comparable to the SQL relational language. XQuery is designed for meeting the requirements of skilled database programmers; its inherent complexity makes the new language unsuited to unskilled users.In this a ...

Keywords: Human interfaces, XML, XQuery, semi-structured data, visual query languages

38 Software engineering: applications, practices and tools (SE): Marshaling and unmarshaling models using the entity-relationship model

Fei Cao, Barrett R. Bryant, Wei Zhao, Carol C. Burt, Rajeev R. Raje, Andrew M. Olson, Mikhail Auguston
March 2005 **Proceedings of the 2005 ACM symposium on Applied computing SAC '05**
Publisher: ACM Press
Full text available: pdf(344.00 KB) Additional Information: full citation, abstract, references, index terms

Software systems are usually designed and documented with the aid of visual modeling notations. Visual modeling notations keep evolving over the years in tandem with visual modeling tools, and the tight binding in between impedes the exchanging of modeling assets, which causes a spatial isolation of the models. Another problem with legacy software models is that they are isolated temporally in the early phases of the software engineering life cycle without reaching out to the later phases. This ...

Keywords: entity-relationship model, marshaling and unmarshaling models, modeling and meta-modeling

39 Data scalability in open hypermedia systems

Kenneth M. Anderson
February 1999 **Proceedings of the tenth ACM Conference on Hypertext and hypermedia : returning to our diverse roots: returning to our diverse roots**
Publisher: ACM Press
Full text available: pdf(1.42 MB) Additional Information: full citation, references, citings, index terms

Keywords: Chimera, open hypermedia systems, scalability

40 Query optimization for OLAP-XML federations

Dennis Pedersen, Karsten Riis, Torben Bach Pedersen
November 2002 **Proceedings of the 5th ACM international workshop on Data Warehousing and OLAP**
Publisher: ACM Press
Full text available: pdf(205.89 KB) Additional Information: full citation, abstract, references, citings, index terms

The changing data requirements of today's dynamic business environments are not handled well by current OLAP systems. Physically integrating unexpected data into such systems is a long and time-consuming process making logical integration, i.e., federation, the better choice in many situations. The increasing use of Extended Markup Language (XML), e.g. in business-to-business (B2B) applications, suggests that the required data will often be available as XML data. This means that logical federati ...

Keywords: OLAP, XML, database federations, query optimization

Results 21 - 40 of 200 Result page: previous 1 **2** 3 4 5 6 7 8 9 10 next

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2006 ACM, Inc.
Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player

PORTAL
USPTO
Subscribe (Full Service) Register (Limited Service, Free) Login
Search: The ACM Digital Library The Guide
query and relationship and class and database and data and tr
SEARCH
THE ACM DIGITAL LIBRARY
Feedback Report a problem Satisfaction survey

Terms used
query and **relationship** and **class** and **database** and **data** and **transfer**

Found **86,991** of **185,942**

Sort results by relevance
Display results expanded form
Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next
Best 200 shown
Relevance scale

1 Extending OLAP querying to external object databases
Torben Bach Pedersen, Arie Shoshani, Junmin Gu, Christian S. Jensen
November 2000 **Proceedings of the ninth international conference on Information and knowledge management**
Publisher: ACM Press
Full text available: pdf(168.32 KB) Additional Information: full citation, references, citings, index terms

2 Fast detection of communication patterns in distributed executions
Thomas Kunz, Michiel F. H. Seuren
November 1997 **Proceedings of the 1997 conference of the Centre for Advanced Studies on Collaborative research**
Publisher: IBM Press
Full text available: pdf(4.21 MB) Additional Information: full citation, abstract, references, index terms

Understanding distributed applications is a tedious and difficult task. Visualizations based on process-time diagrams are often used to obtain a better understanding of the execution of the application. The visualization tool we use is Poet, an event tracer developed at the University of Waterloo. However, these diagrams are often very complex and do not provide the user with the desired overview of the application. In our experience, such tools display repeated occurrences of non-trivial commun ...

3 Computing curricula 2001
September 2001 **Journal on Educational Resources in Computing (JERIC)**
Publisher: ACM Press
Full text available: pdf(613.63 KB) html(2.78 KB) Additional Information: full citation, references, citings, index terms

4 Types and persistence in database programming languages
Malcolm P. Atkinson, O. Peter Buneman
June 1987 **ACM Computing Surveys (CSUR)**, Volume 19 Issue 2
Publisher: ACM Press
Full text available: pdf(7.91 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Traditionally, the interface between a programming language and a database has either been through a set of relatively low-level subroutine calls, or it has required some form of

embedding of one language in another. Recently, the necessity of integrating database and programming language techniques has received some long-overdue recognition. In response, a number of attempts have been made to construct programming languages with completely integrated database management systems. These lang ...

5 An object-oriented approach to multimedia databases

Darrell Woelk, Won Kim, Willis Luther

June 1986 **ACM SIGMOD Record , Proceedings of the 1986 ACM SIGMOD international conference on Management of data SIGMOD '86**, Volume 15 Issue 2

Publisher: ACM Press

Full text available: pdf(1.76 MB) Additional Information: full citation, abstract, references, citings, index terms

This paper identifies data modelling and data access and sharing requirements which multimedia applications impose on a database system. It shows the capabilities of an object-based data model and indicates extensions which are needed to meet the data modelling aspects of these requirements. A logical implementation of the operations on the model is described. The model generalizes the notions of instantiation and generalization in the standard object-oriented paradigm, and augments it with ...

6 Query evaluation techniques for large databases

Goetz Graefe

June 1993 **ACM Computing Surveys (CSUR)**, Volume 25 Issue 2

Publisher: ACM Press

Full text available: pdf(9.37 MB) Additional Information: full citation, abstract, references, citings, index terms, review

Database management systems will continue to manage large data volumes. Thus, efficient algorithms for accessing and manipulating large sets and sequences will be required to provide acceptable performance. The advent of object-oriented and extensible database systems will not solve this problem. On the contrary, modern data models exacerbate the problem: In order to manipulate large sets of complex objects as efficiently as today's database systems manipulate simple records, query-processi ...

Keywords: complex query evaluation plans, dynamic query evaluation plans, extensible database systems, iterators, object-oriented database systems, operator model of parallelization, parallel algorithms, relational database systems, set-matching algorithms, sort-hash duality

7 Data placement in shared-nothing parallel database systems

Manish Mehta, David J. DeWitt

February 1997 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 6 Issue 1

Publisher: Springer-Verlag New York, Inc.

Full text available: pdf(245.08 KB) Additional Information: full citation, abstract, citings, index terms

Data placement in shared-nothing database systems has been studied extensively in the past and various placement algorithms have been proposed. However, there is no consensus on the most efficient data placement algorithm and placement is still performed manually by a database administrator with periodic reorganization to correct mistakes. This paper presents the first comprehensive simulation study of data placement issues in a shared-nothing system. The results show that current hardware techn ...

Keywords: Declustering, Disk allocation, Resource allocation, Resource scheduling

8 The FINITE STRING Newsletter: Abstracts of current literature

Computational Linguistics Staff

January 1987 **Computational Linguistics**, Volume 13 Issue 1-2

Publisher: MIT Press
Full text available: pdf(6.15 MB) Additional Information: full citation
Publisher Site

9 The complexity of querying indefinite data about linearly ordered domains
R. van der Meyden
July 1992 **Proceedings of the eleventh ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems**
Publisher: ACM Press
Full text available: pdf(1.62 MB) Additional Information: full citation, references, citings, index terms

10 Joint workshop on web services modeling and implementation using sound web engineering practices and methods, archhitectures & technologies for e-service engineering (SMIWEP-MATeS'06): Integrating bioinformatic data sources over the SFSU ER design tools XML databus
Yan Liu, Sorna Vincent, Marguerite C. Murphy
July 2006 **Workshop proceedings of the sixth international conference on Web engineering ICWE '06**
Publisher: ACM Press
Full text available: pdf(349.08 KB) Additional Information: full citation, abstract, references, index terms

The SFSU ER Design Tools were developed to support database design and data integration over multiple implementation data models. These tools allow users to enter and view Entity Relationship (ER) schemas and to translate ER schemas into a variety of equivalent implementation schemas, including Relational (ANSI SQL2), Object Oriented (ODMG 3.0), Spreadsheet (Universal Relation with associated functional dependencies) and W3C XML DTD. In addition, for each implementation data model, the Tools gen ...

Keywords: Web services data integration, distributed query processing, join processing, schema integration

11 Special issue: AI in engineering
D. Sriram, R. Joobbani
April 1985 **ACM SIGART Bulletin**, Issue 92
Publisher: ACM Press
Full text available: pdf(8.79 MB) Additional Information: full citation, abstract

The papers in this special issue were compiled from responses to the announcement in the July 1984 issue of the SIGART newsletter and notices posted over the ARPAnet. The interest being shown in this area is reflected in the sixty papers received from over six countries. About half the papers were received over the computer network.

12 Advanced data processing in KRISYS: modeling concepts, implementation techniques, and client/server issues
Stefan Deßloch, Theo Härder, Nelson Mattos, Bernhard Mitschang, Joachim Thomas
May 1998 **The VLDB Journal — The International Journal on Very Large Data Bases**, Volume 7 Issue 2
Publisher: Springer-Verlag New York, Inc.
Full text available: pdf(210.27 KB) Additional Information: full citation, abstract, citings, index terms

The increasing power of modern computers is steadily opening up new application domains for advanced data processing such as engineering and knowledge-based applications. To meet their requirements, concepts for advanced data management have been investigated during the last decade, especially in the field of object orientation. Over the last couple of years, the database group at the University of Kaiserslautern has been developing such an

advanced database system, the KRISYS prototype. In this ...

Keywords: Client/server architectures, Consistency control, Object-oriented modeling concepts, Query processing, Run-time optimization

13 Complete answer aggregates for treelike databases: a novel approach to combine querying and navigation

Holger Meuss, Klaus U. Schulz

April 2001 **ACM Transactions on Information Systems (TOIS)**, Volume 19 Issue 2

Publisher: ACM Press

Full text available: pdf(356.60 KB) Additional Information: full citation, abstract, references, citings, index terms

The use of markup languages like SGML, HTML or XML for encoding the strucutre of documents or linguistic data has lead to many databases where entries are adequately described as trees. In this context querying formalisms are interesting that offer the possiblity to refer both to textual content and logical structure. We consider models where the strucutre specified in a query is not only used as a filter, but also for selecting and presenting different parts of the data. If answers are formaliz ...

Keywords: SGML, XML, answer presentation, information retrieval, logic, query languages, semistructured data, structured documents, tree databases, tree matching

14 Metadata management in federated multimedia systems

Mark Roantree

January 2002 **Australian Computer Science Communications , Proceedings of the thirteenth Australasian database conference - Volume 5 ADC '02**, Volume 24 Issue 2

Publisher: Australian Computer Society, Inc., IEEE Computer Society Press

Full text available: pdf(1.00 MB) Additional Information: full citation, abstract, references, citings, index terms

A Federated Information System requires that multiple (often heterogenous) information systems are integrated to the extent that they can share data through views. One issue faced during the construction of federated view schemata is the continuous need to extract metadata from cooperating systems. This is more pressing in the case of federated multimedia systems where needless transfer of large binary objects affects system performance. Where participating systems employ an object-oriented comm ...

Keywords: ODMG, federated databases, interoperability, metadata, object-oriented databases

15 Federated database systems for managing distributed, heterogeneous, and autonomous databases

Amit P. Sheth, James A. Larson

September 1990 **ACM Computing Surveys (CSUR)**, Volume 22 Issue 3

Publisher: ACM Press

Full text available: pdf(5.02 MB) Additional Information: full citation, abstract, references, citings, index terms, review

A federated database system (FDBS) is a collection of cooperating database systems that are autonomous and possibly heterogeneous. In this paper, we define a reference architecture for distributed database management systems from system and schema viewpoints and show how various FDBS architectures can be developed. We then define a methodology for developing one of the popular architectures of an FDBS. Finally, we discuss critical issues related to developing and operating an FDBS.

16 Model instantiation for query driven simulation in active KDL

John A. Miller, Walter D. Potter, Krys J. Kochut, Orville R. Weyrich
April 1990 **ACM SIGSIM Simulation Digest , Proceedings of the 23rd annual symposium on Simulation ANSS '90**, Volume 20 Issue 4
Publisher: IEEE Press, ACM Press

Full text available: pdf(1.58 MB) Additional Information: full citation, abstract, references, citings, index terms

The focus of this paper is on how Active KDL can be used to provide a very powerful simulation support environment. Active KDL (Knowledge/Data Language) is an object-oriented database programming language, which provides access to integrated model, knowledge, and data bases. Simulation inputs and outputs can be stored by Active KDL since it supports complex objects. More importantly, Active KDL also allows users to specify rules to capture heuristic knowled ...

17 Migrating relational data to an ODBMS: strategics and lessons from a molecular biology experience

Jon Oler, Gary Lindstrom, Terence Critchlow
October 1997 **ACM SIGPLAN Notices , Proceedings of the 12th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications OOPSLA '97**, Volume 32 Issue 10
Publisher: ACM Press

Full text available: pdf(1.40 MB) Additional Information: full citation, abstract, references, index terms

The growing maturity of ODBMS technology is causing many enterprises to consider migrating relational databases to ODBMS's. While data remapping is relatively straightforward in most cases, greater challenges lie in economically and non-invasively adapting legacy application software. We report on a genetics laboratory database migration experiment, which was facilitated by both organization of the relational data in object-like form and a C++ framework designed to insulate application code from ...

18 Join processing in relational databases

Priti Mishra, Margaret H. Eich
March 1992 **ACM Computing Surveys (CSUR)**, Volume 24 Issue 1
Publisher: ACM Press

Full text available: pdf(4.42 MB) Additional Information: full citation, abstract, references, citings, index terms, review

The join operation is one of the fundamental relational database query operations. It facilitates the retrieval of information from two different relations based on a Cartesian product of the two relations. The join is one of the most diffidult operations to implement efficiently, as no predefined links between relations are required to exist (as they are with network and hierarchical systems). The join is the only relational algebra operation that allows the combining of related tuples fro ...

Keywords: database machines, distributed processing, join, parallel processing, relational algebra

19 Multimedia data indexing: Looking at mapping, indexing & querying of MPEG-7 descriptors in RDBMS with SM3

Yang Chu, Liang-Tien Chia, Sourav S. Bhowmick
November 2004 **Proceedings of the 2nd ACM international workshop on Multimedia databases**
Publisher: ACM Press

Full text available: pdf(279.92 KB) Additional Information: full citation, abstract, references, index terms

MPEG-7 documents, which are primarily for multimedia information exchange, are also data-centric XML documents. Due to its advantages, the relational DBMS is the best choice for storing such XML documents. Storing XML data in relational DBMS can be classified into

two classes of storage model: structure-mapping and model-mapping. However, the structure-mapping model cannot support complex Xpath-based query efficiently and model mapping approach lacks the flexible capability in representing al ...

Keywords: MPEG-7, SM3, relational DBMS, storing XML documents

20 Geographic Data Modeling: Fuzzy topological predicates, their properties, and their integration into query languages

Markus Schneider

November 2001 **Proceedings of the 9th ACM international symposium on Advances in geographic information systems**

Publisher: ACM Press

Full text available: pdf(1.22 MB) Additional Information: full citation, abstract, references, index terms

For a long time topological relationships between spatial objects have been a main focus of research on spatial data handling and reasoning. They have especially been integrated into query languages of spatial database systems and geographical information systems. One of their fundamental features is that they operate on spatial objects with precisely defined, sharp boundaries. But in many geometric and geographic applications there is a need to model spatial phenomena and their topological rela ...

Keywords: fuzzy region, fuzzy spatial query language

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2006 ACM, Inc.
Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player

203202

STIC EIC 2100
Search Request Form

136

Today's Date: 28 September 2006 **What date would you like to use to limit the search?**

Priority Date: 7/14/2000 Other:

Name _SHAHID ALAM_____

AU _2162_____ Examiner # 74493_____

Room # __3A41_____ Phone __2-4030____

Serial # ____10//825,417_____

Format for Search Results (Circle One):

PAPER DISK EMAIL

Where have you searched so far?

USP DWPI EPO JPO ACM IBM TDB

IEEE INSPEC SPI Other _____

Is this a "Fast & Focused" Search Request? (Circle One) **YES** **NO**

A "Fast & Focused" Search is completed in 2-3 hours (maximum). The search must be on a very specific topic and meet certain criteria. The criteria are posted in EIC2100 and on the EIC2100 NPL Web Page at http://ptoweb/patents/stic/stic-tc2100.htm.

What is the topic, novelty, motivation, utility, or other specific details defining the desired focus of this search? Please include the concepts, synonyms, keywords, acronyms, definitions, strategies, and anything else that helps to describe the topic. Please attach a copy of the abstract, background, brief summary, pertinent claims and any citations of relevant art you have found.

A home contains information about the table, including its physical name and qualifier, the physical names of its columns, and which columns are key columns. An object oriented method for implementing a persistence mechanism adapted to effectuate data transfer between a application and a relational database, identifying homes and relationships, each identified home comprising attributes of a table of the database, each identified relationship having attributes describing an interaction between two homes; a datastore is associated to the database, including attributes of the identified homes and relationships; generating a query to transfer data between the application and the database; generating a text document, the text document includes attributes of the datastore and the text document is a extensible Markup Language (XML) document.

I am enclosing Background, summary and claims and EAST search history for the review.

STIC Searcher Emory Phone 2-3520

Date picked up 9/28/6 Date Completed 9/29/6

sira Search and Information Resources Administration

XCOPY BM

```
Set     Items   Description
S1     108231   S JAVA? OR XML OR EXTENS?()(MARKUP? OR MARK()UP)()LANGUAG? OR QXML OR SGML
S2       6646   S JAVASCRIPT? OR JAVA()SCRIPT? OR APPLET OR APPLETS OR SERVLET? OR XRML?
S3    3051350   S TRANSFER? OR RELOCAT? OR MIGRAT? OR TRANSLOCAT?
S4    2028038   S MIGRAT? OR SENT OR MOVE? OR MOVING? OR ROUTING?
S5    3782424   S SHIP???? OR MAIL???? OR ROUTE? OR MAP? ? OR MAPP???? OR REROUT? OR
CONNECT?
S6     429946   S REDIRECT? OR TRANSMIGRAT? OR DISPATCH? OR UPLOAD? OR DOWNLOAD? OR
FORWARD?
S7    5887224   S IMPORT? OR EXPORT? OR TRANSPORT?
S8    7987925   S DATA?
S9   10942084   S FILE? ? OR DATAFILE? OR PROGRAM? OR APPLICATION? OR SOFTWARE? OR MACRO?
OR SCRIPT?
S10         0   S DATARECORD? ? OR DATA()RECORD?OR DATA()FILE?
S11    195872   S CONTROL()LOGIC? OR EXECUT?()FILE? OR SOFT()WARE? OR PROGRAM?()LANGUAG?
S12   8967779   S SOFTWARE? OR APPLICATION? OR SOFT()WARE? OR APP? ? OR MACRO? ? OR
INTERFACE?
S13    711292   S SUBROUTIN? OR SUBPROGRAM? OR COMPUTER?(2N)(CODE? OR UTILIT? OR SCRIPT?
OR PROGRAM? OR ROUTINE? OR SUBROUTIN? OR LANGUAG?)
S14     54186   S (CPU? ? OR PROCESSOR?)(2N)(PROGRAM? OR APPLICATION? OR ROUTINE? OR
SUBROUTINE? OR CODE? OR INSTRUCTION? OR ALGORITH? OR LANGUAG?)
S15    154817   S (CPU? ? OR PROCESSOR? OR COMPUTER?)(2N)(PROCEDUR? OR PROTOCOL? OR
COMMUNICATION? OR COMMAND? OR STEP? ?)
S16   1034466   S COMPUTER(2N)(SCRIPT? OR PROGRAM? OR APPLICATION? OR MACRO? ? OR UTILIT?)
S17    898979   S DATABASE? OR RDB OR RDBS OR RDBMS OR DATA()STRUCTUR? OR DATA()BASE?
S18    218035   S DATA()(BASE? OR FILE? OR BANK? OR REGIST? OR REPOSIT? OR ARCHIV?)
S19     45383   S RELATIONAL?()(DATABASE? OR DATA()BASE? OR DB OR DBS OR DBMS) OR RDB OR
RDBS OR RDBMS
S20     12727   S (ORACLE? OR PARADOX? OR SQL OR HQL OR QUERY?()LANGUAGE?)(3N)(DATABASE?
OR DATA()BASE? OR DB OR DBS OR DBMS)
S21     45915   S RELATIONAL?(3N)(DB OR DBS OR DATABASE? OR DATA()BASE?) OR RDB OR RDBS
S22        49   S RELATIONAL?(3N)(DATAFILE? OR DATASTOR? OR DATABANK?)
S23       482   S RELATIONAL?(3N)DATA()(FILE? ? OR STORE? OR STORAG? OR STORING? OR BANK?
OR REPOSIT? OR DEPOSITOR? OR SERVER?)
S24      1653   S (R OR RELATIONAL?)()(DBMS? OR MANAG?()SYSTEM?)
S25      6781   S S1:S2 AND S3:S7 AND S8:S16 AND S17:S24
S26       443   S ATTRIBUT? OR PRIMITIV? OR INTEGER? OR STRING? OR OBJECT()(REFERENC? OR
POINTER?)
S27      1217   S METADATA? OR META()DATA? OR RELATIONSHIP? OR INTERACTION?
S28        14   S (HOME OR TABLE? OR COLUMN? OR KEYCOLUMN?)(3N)(INFORMATION? OR NAME? OR
QUALIFIER?)
S29         8   S (HOME OR TABLE? OR COLUMN? OR KEYCOLUMN?)(3N)(ATTRIBUT? OR POINTER? OR
IDENTIFIER?)
S30      1240   S FIRST? OR 1ST OR PRIMARY OR INITIAL? OR ORIGINAL?
S31       516   S LEADOFF? OR MAIN OR CHIEF OR INTRODUCTORY? OR MASTER?
S32        37   S NUMBER()(ONE OR 1) OR PRINCIPAL?
S33      2212   S TEXT? OR DOCUMENT? OR TEXTDOCUMENT? OR XMLDOCUMENT? OR JAVADOCUMENT?
S34       115   S STYLE()SHEET? OR STYLESHEET? OR SHEET? OR SINGLEPAGE?
S35      1138   S FORM? ? OR PAGE? ? OR THUMB()NAIL? OR PREVIEW?
S36       568   S REQUEST? OR INTERROGAT? OR FETCH? OR PREFETCH?
S37      2417   S QUERY? OR QUERIE? OR DATAMIN? OR DATA()(MINE? OR MINING) OR SEARCH? OR
SEARCH()ENGINE?
S38        18   S KEYWORDSEARCH? OR TEXTMIN? OR TEXT()(MINE? OR MINING)
S39       268   S ASK OR ASKS OR ASKED OR ASKING OR INQUIR? OR INTERROGAT? OR CRAWL? OR
DATACRAWL?
S40      1478   S FIND? OR LOCATE? OR LOCATING OR RESEARCH? OR LOOK?()UP OR LOOKUP?
S41       798   S BEGIN? OR BEGUN? OR START? OR COMMENC? OR TRIGGER? OR EXECUT? OR
INITIALIZ? OR INITIALIS?
S42       214   S LAUNCH? OR ACTIVAT? OR (CARRY? OR CARRIE?)()OUT OR (SET OR SETS OR
SETTING)(2W)MOTION
S43      1411   S INSTITUT? OR PRODUCE? OR PRODUCING OR GENERAT?
```

```
S44        255   S PERSIST? OR PERMANEN? OR DEDICATED?
S45        165   S S25 AND S41:S43(5N)S33:S35(5N)S1:S2
S46       3069   S S25 AND S3:S7(5N)S8:S10 AND S3:S8(7N)(S1:S2 OR S11:S16) AND
S3:S8(7N)S17:S24
S47         79   S S45 AND S46
S48        684   S S46 AND S26:S29
S49         25   S S48 AND S44
S50         12   S S48 AND S41:S43(5N)S36:S40(5N)S3:S7(7N)S8:S10 AND (S11:S16 OR S1:S2) AND
S17:S24
S51        201   S S45 OR S47 OR S49:S50
S52         68   S S51 AND S30:S32
S53        201   S S51:S52
S54        702   S S46 AND S1:S2(5N)S33:S35
S55         75   S S54 AND (S47 OR S49:S53)
S56        201   S S53 OR S55
S57         59   S S56 AND PY=1970:2000
S58         60   S S56 NOT PY=2001:2006
S59         60   S S57:S58
S60         56   RD  (unique items)
 ; show files
```

[File 2] **INSPEC** 1898-2006/Sep W3
(c) 2006 Institution of Electrical Engineers. All rights reserved.

[File 6] **NTIS** 1964-2006/Sep W3
(c) 2006 NTIS, Intl Cpyrght All Rights Res. All rights reserved.

[File 8] **Ei Compendex(R)** 1970-2006/Sep W3
(c) 2006 Elsevier Eng. Info. Inc. All rights reserved.

[File 34] **SciSearch(R) Cited Ref Sci** 1990-2006/Sep W4
(c) 2006 The Thomson Corp. All rights reserved.

[File 35] **Dissertation Abs Online** 1861-2006/Sep
(c) 2006 ProQuest Info&Learning. All rights reserved.

[File 56] **Computer and Information Systems Abstracts** 1966-2006/Sep
(c) 2006 CSA. All rights reserved.

[File 60] **ANTE: Abstracts in New Tech & Engineer** 1966-2006/Sep
(c) 2006 CSA. All rights reserved.

[File 65] **Inside Conferences** 1993-2006/Sep 28
(c) 2006 BLDSC all rts. reserv. All rights reserved.

[File 94] **JICST-EPlus** 1985-2006/Jun W3
(c)2006 Japan Science and Tech Corp(JST). All rights reserved.

[File 99] **Wilson Appl. Sci & Tech Abs** 1983-2006/Jul
(c) 2006 The HW Wilson Co. All rights reserved.

[File 111] **TGG Natl.Newspaper Index(SM)** 1979-2006/Sep 15
(c) 2006 The Gale Group. All rights reserved.

[File 144] **Pascal** 1973-2006/Sep W1
(c) 2006 INIST/CNRS. All rights reserved.

[File 239] **Mathsci** 1940-2006/Nov
(c) 2006 American Mathematical Society. All rights reserved.

[File 256] **TecInfoSource** 82-2006/Jan
(c) 2006 Info.Sources Inc. All rights reserved.

60/3,K/7 (Item 7 from file: 2) **Links**
INSPEC
(c) 2006 Institution of Electrical Engineers. All rights reserved.
07636414 **INSPEC Abstract Number:** C2000-08-6150E-004
Title: A generic load/extract utility for data transfer between XML documents and relational databases
Author Bourret, R.; Bornhovd, C.; Buchmann, A.
Author Affiliation: Dept. of Comput. Sci., Tech. Hochschule Darmstadt, Germany
Conference Title: Proceedings Second International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems. WECWIS 2000 p. 134-43
Editor(s): Yu, P.S.
Publisher: IEEE Comput. Soc , Los Alamitos, CA, USA
Publication Date: 2000 **Country of Publication:** USA ix+255 pp.
ISBN: 0 7695 0610 0 **Material Identity Number:** XX-2000-01455
U.S. Copyright Clearance Center Code: 0 7695 0610 0/2000/$10.00
Conference Title: Proceedings Second International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems. WECWIS 2000
Conference Sponsor: Univ. California, Irvine; Univ. Virginia; IEEE Comput. Soc. Tech. Committee on Real-Time Syst.; IEEE Comput. Soc. Tech. Committee on Internet; IBM Inst. Adv. Commerce; Univ. Hong Kong
Conference Date: 8-9 June 2000 **Conference Location:** Milpitas, CA, USA
Language: English
Subfile: C
Copyright 2000, IEE
Identifiers: ...**data transfer**;**XML documents**;**relational databases**; semantics-preserving **data** exchange format... ...structured **data** extraction... ...**data** loading... ...**mapping** language
2000

60/3,K/5 (Item 5 from file: 2) **Links**
INSPEC
(c) 2006 Institution of Electrical Engineers. All rights reserved.
07714868 **INSPEC Abstract Number:** C2000-11-6115-004
Title: Hype-a Java tool for the rapid development of hyperdocument management applications for the WWW

Author Scholz, E.; Lischetzki, R.
Author Affiliation: DaimlerChrysler Res. & Technol., Berlin, Germany
Conference Title: JIT'99. Java-Information-Tag 1999 p. 85-99
Editor(s): Cap, C.H.
Publisher: Springer-Verlag , Berlin, Germany
Publication Date: 1999 **Country of Publication:** Germany xi+296 pp.
ISBN: 3 540 66464 5 **Material Identity Number:** XX-1999-02729
Conference Title: Proceedings of JIT'99: Java-Informations-Tage
Conference Date: 20-21 Sept. 1999 **Conference Location:** Dusseldorf, Germany
Language: English
Subfile: C
Copyright 2000, IEE
Title: Hype-a Java tool for the rapid development of hyperdocument management applications for the WWW

Abstract: The paper presents Hype, an integrated development environment for building typed hyperdocument management applications for the World Wide Web. Like Java Server Pages, Hype enables Java code snippets producing dynamic content to be embedded into static HTML pages. Unlike Java Server Pages, Hype covers the complete process of building three-tier applications. This includes the object oriented modeling of the application schema in a graphical editor, the mapping of the object oriented application schema to a relational database schema, and the automatic generation of persistent Java classes for accessing the database. Knowing the application schema, Hype can go well beyond Java Server Pages with regard to the typesafeness and modularity it provides for producing dynamic content. Unlike many commercial tool suites, Hype is not targeted mainly at software professionals. Rather, the design goal of Hype was to enable power users, i.e., application domain experts with limited programming expertise, to implement most of a hyperdocument management application without programming. An important means for this (and a hallmark of Hype) is a small set of typed placeholders, which replace the use of embedded Java for many common customization tasks.
Descriptors: ...Java; programming environments
Identifiers: ...Java tool... ...hyperdocument management application development... ...typed hyperdocument management applications; Java Server Pages; Java code snippets... ...three-tier applications; object oriented application schema... ...relational database schema... ...persistent Java classes... ...software professionals... ...application domain experts
1999

60/3,K/10 (Item 10 from file: 2) **Links**
INSPEC
(c) 2006 Institution of Electrical Engineers. All rights reserved.
07559317 **INSPEC Abstract Number:** C2000-05-7840-029
Title: An implementation of XML and JAVA in a prototypal environment. Standard for satellite information production and dissemination
Author Colaiacomo, L.; Valero, J.L.; Mariano, M.L.; Del Giudice, R.; Ferri, P.
Conference Title: DASIA 99. Data Systems in Aerospace (ESA SP-447) p. 549-51
Publisher: ESA , Noordwijk, Netherlands
Publication Date: 1999 **Country of Publication:** Netherlands 644 pp.
Material Identity Number: XX-2000-00519
Conference Title: DASIA 99. Data Systems in Aerospace
Conference Date: 17-21 May 1999 **Conference Location:** Lisbon, Portugal
Language: English
Subfile: C
Copyright 2000, IEE
Title: An implementation of XML and JAVA in a prototypal environment. Standard for satellite information production and dissemination
Abstract: ...Satellite Centre (located in Torrejon de Ardoz, Madrid, Spain), a prototype to arrange geographic information data (GIS data) and collateral information into extended Markup Language (XML) format documents has been developed by Cap Gemini Italia SpA. The prototype implementing the above is named DWARF (Data Warehouse Archiving and Reporting Facility), and is based on Web technologies to have the maximum... ...the XT4L document) that can be exploited from commercial browsers without having the need to download particular software, and giving enough flexibility to exploit all the information contained in the document itself. It was recognised from the beginning that the XML standard was powerful enough to satisfy the requirements of the portability and of the flexibility... ...information produced in the WEUSC on distributed platforms and architecture. Moreover because of the different data types, XML and Java give an interface to handle in a common environment such different sources.
Descriptors: data structures; Java; software portability... ...software prototyping... ...software reusability
Identifiers: XML; JAVA; geographic information data; GIS data; Data Warehouse Archiving and Reporting Facility... ...XML standard... ...data types
1999

60/3,K/11 (Item 11 from file: 2) **Links**

Fulltext available through: Ex Libris USPTO Full Text Retrieval Options SCIENCEDIRECT

INSPEC

(c) 2006 Institution of Electrical Engineers. All rights reserved.

07455191 **INSPEC Abstract Number:** C2000-02-6130D-009

Title: Getting started with XSLT style sheets and Java servlets

Author Dumbill, E.

Journal: WEB Techniques vol.4, no.12 p. 77-80, 82-3

Publisher: Miller Freeman ,

Publication Date: Dec. 1999 **Country of Publication:** USA

CODEN: WETEFA **ISSN:** 1086-556X

SICI: 1086-556X(199912)4:12L.77:GSWX;1-O

Material Identity Number: F184-1999-011

Language: English

Subfile: C

Copyright 2000, IEE

Title: Getting started with XSLT style sheets and Java servlets

Abstract: The world of XSL is moving fast, and developers want to know how this technology can benefit them. The XSL Transformations... ...in parallel with the Working Draft. We're already seeing XSLT capabilities built in to application and database servers, and it won't be long before XSL capabilities will be part of these... ...own" XSL-based Web server, and you'll gain some valuable experience with XSL and servlets while you're at it. The article assumes you have a basic familiarity with Java, XML, and XSLT. The combination of XML and XSLT is a powerful one. Documents can be created in XML, and the HTML presentation can be derived automatically from the original data. This mechanism gives you all the advantages associated with separating presentation from content, notably an easily changeable appearance, and preservation of the original structured content. However, you could accomplish this with Make and Perl, and indeed some developers have been doing it this way for several years. When we add servlets to the equation we can generate parts, or the whole, of our XML document dynamically to produce personalized, database-backed, or otherwise adaptive pages while still maintaining the advantages of separating content and style.

Descriptors: ...file servers... ...Java

Identifiers: ...Java servlets; database servers... ...XML document

1999

60/3,K/15 (Item 15 from file: 2) **Links**
INSPEC
(c) 2006 Institution of Electrical Engineers. All rights reserved.
07160579 **INSPEC Abstract Number:** C1999-03-6160B-015
Title: Defining reusable, distributable information objects using SGML: or, How SGML can do for databases what Java has done for user interfaces
Author Chelsom, J.
Author Affiliation: CSW Inf. Ltd., UK
Conference Title: SGML Europe '97. Conference Proceedings p. 307-9
Publisher: Graphic Commun. Assoc , Alexandria, VA, USA
Publication Date: 1996 **Country of Publication:** USA 341 pp.
Material Identity Number: XX-1997-00997
Conference Title: Proceedings of SGML '97. The Next Decade - Pushing the Envelope
Conference Date: 13-15 May 1997 **Conference Location:** Barcelona, Spain
Language: English
Subfile: C
Copyright 1999, IEE
Title: Defining reusable, distributable information objects using SGML: or, How SGML can do for databases what Java has done for user interfaces
Abstract: Just as Java has brought an open, distributable way of enabling users to interact with data by transferring applications in real time from server to client, so SGML can enable them to interact with persistent database objects by transferring, in real time, the database schema for those objects. This paper explores the potential of SGML as a universal database definition language for reusable, distributable information objects and shows how existing technology is already turning...
Descriptors: database languages... ...distributed databases; electronic data interchange... ...Java; object-oriented databases; persistent objects... ...software reusability
Identifiers: ...SGML; distributed databases; Java; user interfaces; data interaction; real-time application transfer; persistent database objects... ...real-time database schema transfer; universal database definition language
1996

60/3,K/26 (Item 5 from file: 8) **Links**
Ei Compendex(R)
(c) 2006 Elsevier Eng. Info. Inc. All rights reserved.
04436305 **E.I. No:** EIP96073229697
Title: Automatically generating Java documentation

Author: Aitken, Gary
Source: Dr. Dobb's Journal of Software Tools for Professional Programmer v 21 n 7 Jul 1996. 6pp
Publication Year: 1996
CODEN: DDJTEQ
Language: English
Title: Automatically generating Java documentation
Abstract: Java doesn't require a header file, hence documentation is really important in a Java environment. This article examines some of the documentation issues related to Java programming and discusses how to automatically generate documentation for Java classes using javadoc.
Descriptors: *Computer aided software engineering; Program documentation; C (programming language); File organization; Codes (symbols); Computational linguistics; Data structures; User interfaces; Program compilers; Algorithms
Identifiers: Java documentation; DOC comment; Syntax; Source file

```
Set     Items   Description
S1      10783   S JAVA? OR XML OR EXTENS?()(MARKUP? OR MARK()UP)()LANGUAG? OR QXML OR SGML
S2       2228   S JAVASCRIPT? OR JAVA()SCRIPT? OR APPLET OR APPLETS OR SERVLET? OR XRML?
S3    1184377   S TRANSFER? OR RELOCAT? OR MIGRAT? OR TRANSLOCAT?
S4    2905913   S MIGRAT? OR SENT OR MOVE? OR MOVING? OR ROUTING?
S5    4580765   S SHIP???? OR MAIL???? OR ROUTE? OR MAP? ? OR MAPP???? OR REROUT? OR
CONNECT?
S6     408408   S REDIRECT? OR TRANSMIGRAT? OR DISPATCH? OR UPLOAD? OR DOWNLOAD? OR
FORWARD?
S7    1278125   S IMPORT? OR EXPORT? OR TRANSPORT?
S8    1969324   S DATA?
S9    1855304   S FILE OR DATAFILE? OR PROGRAM? OR APPLICATION? OR SOFTWARE? OR MACRO? OR
SCRIPT?
S10         2   S DATARECORD? ? OR DATA()RECORD?OR DATA()FILE?
S11     11977   S CONTROL()LOGIC? OR EXECUT?()FILE? OR SOFT()WARE? OR PROGRAM?()LANGUAG?
S12   1477786   S SOFTWARE? OR APPLICATION? OR SOFT()WARE? OR APP? ? OR MACRO? ? OR
INTERFACE?
S13    103659   S SUBROUTIN? OR SUBPROGRAM? OR COMPUTER?(2N)(CODE? OR UTILIT? OR SCRIPT?
OR PROGRAM? OR ROUTINE? OR SUBROUTIN? OR LANGUAG?)
S14     35197   S (CPU? ? OR PROCESSOR?)(2N)(PROGRAM? OR APPLICATION? OR ROUTINE? OR
SUBROUTINE? OR CODE? OR INSTRUCTION? OR ALGORITH? OR LANGUAG?)
S15     55702   S (CPU? ? OR PROCESSOR? OR COMPUTER?)(2N)(PROCEDUR? OR PROTOCOL? OR
COMMUNICATION? OR COMMAND? OR STEP? ?)
S16    274002   S COMPUTER(2N)(SCRIPT? OR PROGRAM? OR APPLICATION? OR MACRO? ? OR UTILIT?)
S17    214470   S DATABASE? OR RDB OR RDBS OR RDBMS OR DATA()STRUCTUR? OR DATA()BASE?
S18    102597   S DATA()(BASE? OR FILE? OR BANK? OR REGIST? OR REPOSIT? OR ARCHIV?)
S19      3775   S RELATIONAL?()(DATABASE? OR DATA()BASE? OR DB OR DBS OR DBMS) OR RDB OR
RDBS OR RDBMS
S20       756   S (ORACLE? OR PARADOX? OR SQL OR HQL OR QUERY?()LANGUAGE?)(3N)(DATABASE?
OR DATA()BASE? OR DB OR DBS OR DBMS)
S21      3869   S RELATIONAL?(3N)(DB OR DBS OR DATABASE? OR DATA()BASE?) OR RDB OR RDBS
S22        17   S RELATIONAL?(3N)(DATAFILE? OR DATASTOR? OR DATABANK?)
S23       252   S RELATIONAL?(3N)DATA()(FILE? ? OR STORE? OR STORAG? OR STORING? OR BANK?
OR REPOSIT? OR DEPOSITOR? OR SERVER?)
S24        36   S (R OR RELATIONAL?)()(DBMS? OR MANAG?()SYSTEM?)
S25      1563   S S1:S2 AND S3:S7 AND S8:S16 AND S17:S24
S26       218   S ATTRIBUT? OR PRIMITIV? OR INTEGER? OR STRING? OR OBJECT()(REFERENC? OR
POINTER?)
S27       267   S METADATA? OR META()DATA? OR RELATIONSHIP? OR INTERACTION?
S28        31   S (HOME OR TABLE? OR COLUMN? OR KEYCOLUMN?)(3N)(INFORMATION? OR NAME? OR
QUALIFIER?)
S29        24   S (HOME OR TABLE? OR COLUMN? OR KEYCOLUMN?)(3N)(ATTRIBUT? OR POINTER? OR
IDENTIFIER?)
S30       503   S FIRST? OR 1ST OR PRIMARY OR INITIAL? OR ORIGINAL?
S31        64   S LEADOFF? OR MAIN OR CHIEF OR INTRODUCTORY? OR MASTER?
S32         9   S NUMBER()(ONE OR 1) OR PRINCIPAL?
S33       606   S TEXT? OR DOCUMENT? OR TEXTDOCUMENT? OR XMLDOCUMENT? OR JAVADOCUMENT?
S34        53   S STYLE()SHEET? OR STYLESHEET? OR SHEET? OR SINGLEPAGE?
S35       528   S FORM? ? OR PAGE? ? OR THUMB()NAIL? OR PREVIEW?
S36       515   S REQUEST? OR INTERROGAT? OR FETCH? OR PREFETCH?
S37       415   S QUERY? OR QUERIE? OR DATAMIN? OR DATA()(MINE? OR MINING) OR SEARCH? OR
SEARCH()ENGINE?
S38         1   S KEYWORDSEARCH? OR TEXTMIN? OR TEXT()(MINE? OR MINING)
S39        50   S ASK OR ASKS OR ASKED OR ASKING OR INQUIR? OR INTERROGAT? OR CRAWL? OR
DATACRAWL?
S40       216   S FIND? OR LOCATE? OR LOCATING OR RESEARCH? OR LOOK?()UP OR LOOKUP?
S41       569   S BEGIN? OR BEGUN? OR START? OR COMMENC? OR TRIGGER? OR EXECUT? OR
INITIALIZ? OR INITIALIS?
S42       118   S LAUNCH? OR ACTIVAT? OR (CARRY? OR CARRIE?)()OUT OR (SET OR SETS OR
SETTING)(2W)MOTION
S43       637   S INSTITUT? OR PRODUCE? OR PRODUCING OR GENERAT?
```

```
S44          86    S PERSIST? OR PERMANEN? OR DEDICATED?
S45        1393    S IC=G06F?
S46        1460    S MC=T01?
S47         710    S S25 AND S3:S7(5N)S8:S10 AND S3:S10(10N)(S1:S2 OR S11:S16)(10N)S17:S24
S48         702    S S47 AND S44:S46
S49         710    S S47:S48
S50         234    S S49 AND S33:S35(7N)S1:S2
S51         230    S S49 AND S26:S29
S52         370    S S50:S51
S53          56    S S52 AND S41:S43(7N)S36:S40 AND S36:S43(7N)S3:S7
S54          62    S S52 AND S30:S31(7N)(S26:S29 OR S17:S24)
S55         169    S S49 AND S1:S2(5N)S33:S35 AND S3:S7(5N)S8:S10 AND (S11:S16 OR
S1:S2)(7N)S17:S24
S56          46    S S55 AND S26:S29 AND S36:S40
S57         133    S S53:S54 OR S56
S58         108    S S57 AND AC=US/PR
S59          49    S S58 AND AY=(1970:2000)/PR
S60          31    S S58 NOT AY=(2001:2006)/PR
S61          25    S S57 NOT S58
S62           6    S S61 AND PY=1970:2000
S63           0    S S61 NOT PY=2001:2006
S64          55    S S59:S60 OR S62
S65          55    IDPAT (sorted in duplicate/non-duplicate order)

? show files
```

[File 347] **JAPIO** Dec 1976-2005/Dec(Updated 060404)
(c) 2006 JPO & JAPIO. All rights reserved.

[File 350] **Derwent WPIX** 1963-2006/UD=200661
(c) 2006 The Thomson Corporation. All rights reserved.
**File 350: DWPI has been enhanced to extend content and functionality of the database. For more info, visit http://www.dialog.com/dwpi/.*

65/7/24 (Item 24 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

0012780138 *Drawing available*
WPI Acc no: 2002-634771/200268
XRPX Acc No: N2002-501355
Multi-tier client/server system comprises clients having browsers for processing documents and maintaining connectivity with relational database servers and for executing application logic
Patent Assignee: SARKAR S S (SARK-I)
Inventor: SARKAR S S

Patent Family (1 patents, 1 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
US 6418448	B1	20020709	US 1999455422	A	19991206	200268	B

ON: 545?

Priority Applications (no., kind, date): US 1999455422 A 19991206

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
US 6418448	B1	EN	39	20		

Alerting Abstract US B1
NOVELTY - Several clients have browsers for processing **documents** in XML and RDF, for creating and maintaining thin client windows on demand for persistent **connectivity** through the internet. **Relational database** servers with **application** logic in form of object packages comprise user defined packages and method and operator **interfaces**. The object request broker services execute **application** logic in CORBA.
DESCRIPTION - The **relational database** servers comprise user defined packages for providing call specifications for set of **interfaces** to embed in SQL queries, for specifying operations over attribute values from multiple tables, for specifying **interfaces** where parameter type definition **maps** to another **interface** or to tables, and user defined packages where uniform resource identifiers are used to **locate** elements in schema objects.
USE - Multi-tier client server system for navigating, **querying** and manipulating information using specifications in resource description frame work and supporting multiple object **relational database** resources over the web.
ADVANTAGE - **Triggers queries** for transactions through thin client windows for **persistent** communication with remote **databases**.
DESCRIPTION OF DRAWINGS - The figure shows the block diagram illustrating a single SQL **query** made over the **relational database** schema along with legacy and existing central **databases**.

Title Terms /Index Terms/Additional Words: MULTI; TIER; CLIENT; SERVE; SYSTEM; COMPRISE; PROCESS; DOCUMENT; MAINTAIN; **CONNECT**; RELATED; **DATABASE**; EXECUTE; APPLY; LOGIC

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-017/30			Main		"Version 7"

US Classification, Issued: 707104100, 707001000, 707100000, 707101000, 707103000, 709229000, 709203000

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): **T01-J05B4B**; **T01-J05B4C**; **T01-N02A3C**; **T01-N03A1**; **T01-N03B2**

Original Publication Data by Authority

United States

Publication No. US 6418448 B1 (Update 200268 B)
Publication Date: 20020709
METHOD AND APPARATUS FOR PROCESSING MARKUP LANGUAGE SPECIFICATIONS FOR DATA AND METADATA USED INSIDE MULTIPLE RELATED INTERNET DOCUMENTS TO NAVIGATE, QUERY AND MANIPULATE INFORMATION FROM A PLURALITY OF OBJECT RELATIONAL DATABASES OVER THE WEB.
Assignee: Sarkar, Shyam Sundar, San Mateo, CA, US (SARK-I)
Inventor: Sarkar, Shyam Sundar, San Mateo, CA, US
Language: EN (39 pages, 20 drawings)
Application: US 1999455422 A 19991206 (Local application)
Original IPC: G06F-17/30(A)
Current IPC: G06F-17/30(A)
Original US Class (main): 707104.1
Original US Class (secondary): 7071 707100 707101 707103 709229 709203
Original Abstract: The present invention provides a system for navigation through multiple documents in Extensible Markup Language and Resource Description Framework to inspect data/metadata in order to either start a transaction on selected item(s)in separate thin client window(s) with persistent connectivity through Internet Inter ORB Protocol or implicitly trigger read-only queries in Structured Query Language (SQL) represented in Resource Description Framework against a unified virtual Database defined over multiple physical disparate object relational databases over the web. An implicitly generated query retrieves desired sets of properties and entities presented in documents of Extensible Markup Language and Resource Description Framework for further navigation. Container types in Resource Description Framework are mapped by this invention to record and table types in a normalized relational model where URIs locating elements in relational schema components over the web are stored as primary keys/foreign keys in normalized tables. Methods and operators on such web objects are defined as part of user-defined package definitions in object relational schema where object request brokers apply such methods or operators on result sets from relational operations anywhere on the web. This invention uniquely incorporates two distinct stages of SQL computations for a collaborative method of preparation, execution and resolution of an object SQL query over disparate locations of multiple object relational databases on the web.
Claim:

1. In a network of computers over the internet, a multi-tier client/server system comprising of:

- A) Clients with browsers capable of processing documents in Extensible Markup Language (XML) and Resource Description Framework (RDF) carrying data/metadata information for electronic commerce and electronic business where these clients make synchronous communications through Hypertext Transfer Protocol (HTTP) with various internet sites and with object relational database servers at multiple locations consisting of XML/RDF integration software and software compliant with Common Object Request Broker Architecture;
- B) Clients with browsers capable of creating and maintaining thin client windows on demand for persistent connectivity through Internet Inter ORB Protocol (IIOP) with remote object relational databases over the web for performing transactions;
- C) Relational database servers with schema and business application logic defined in the form of object packages comprising:
- a) User-defined packages for providing call specifications for a set of interfaces to embed in Structured Query Language (SQL) queries where these interfaces are implemented as methods in Java classes and are applied to records resulting from execution of relational operations;
- b) User-defined packages for specifying operations over attribute values from multiple tables where interface definitions for methods and operators carry parameter definitions along with parameter types defined directly as table names and attribute names for safe execution semantics;
- c) User-defined packages specifying interfaces where a parameter type definition maps to another interface existing at a local or remote schema object on the internet;
- d) User-defined packages specifying interfaces where parameter type definitions map to tables and attributes existing at a local or remote schema object on the internet;
- e) User-defined packages where Uniform Resource Identifiers (URIs) are used to locate elements in remote schema objects on the web and such URIs defining schema objects are stored in data dictionaries or repositories;
- f) Method and operator interfaces defined in user-defined packages implemented in Java classes;
- D) Executions of application logic in object packages on local or remote schema objects performed by Object Request Broker services in Common Object Request Broker Architecture (CORBA).

US006418448B1

(12) **United States Patent**
Sarkar

(10) **Patent No.: US 6,418,448 B1**
(45) **Date of Patent: Jul. 9, 2002**

(54) **METHOD AND APPARATUS FOR PROCESSING MARKUP LANGUAGE SPECIFICATIONS FOR DATA AND METADATA USED INSIDE MULTIPLE RELATED INTERNET DOCUMENTS TO NAVIGATE, QUERY AND MANIPULATE INFORMATION FROM A PLURALITY OF OBJECT RELATIONAL DATABASES OVER THE WEB**

(76) Inventor: **Shyam Sundar Sarkar**, 273 Lauric Meadows Dr. Apt. #142, San Mateo, CA (US) 94403

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/455,422**

(22) Filed: **Dec. 6, 1999**

(51) **Int. Cl.**[7] **G06F 17/30**

(52) **U.S. Cl.** **707/104.1**; 707/1; 707/100; 707/101; 707/103; 709/229; 709/203

(58) **Field of Search** 707/104.1, 103, 707/100, 101, 1; 709/203, 229

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,561,795	A *	10/1996	Sarkar	707/202
5,687,361	A *	11/1997	Sarkar	707/1
5,737,592	A	4/1998	Nguyen et al.	707/4
5,878,419	A	3/1999	Carter et al.	707/10
5,893,118	A	4/1999	Sonderegger et al.	707/203
5,895,465	A	4/1999	Guha et al.	707/4
6,012,067	A	1/2000	Sarkar	707/103
6,044,466	A *	3/2000	Anand et al.	713/200
6,145,738	A *	11/2000	Stinson et al.	235/379
6,177,932	B1 *	1/2001	Galdes et al.	345/329
2001/0039563	A1 *	11/2001	Tian	709/202

OTHER PUBLICATIONS

Swick, Ralph R., "Putting It Together: RDF:Weaving the Web of Discovery," ACM, pp. 21–25, Jun. 1999.*

Lee et al. "Information Integration with Attribution Support for Corporate Profiles," ACM, pp. 423–429, 1999.*

"Developing Stored Procedures in Java"—An Oracle Technical White Paper, Apr. 1999.

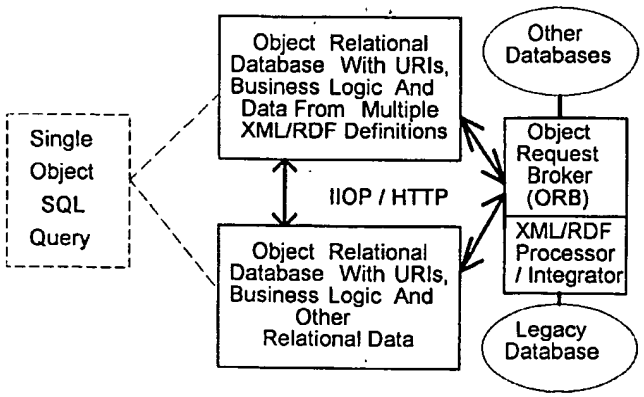
(List continued on next page.)

Primary Examiner—Charles L. Rones

(57) **ABSTRACT**

The present invention provides a system for navigation through multiple documents in Extensible Markup Language and Resource Description Framework to inspect data/metadata in order to either start a transaction on selected item(s) in separate thin client window(s) with persistent connectivity through Internet Inter ORB Protocol or implicitly trigger read-only queries in Structured Query Language (SQL) represented in Resource Description Framework against a unified virtual Database defined over multiple physical disparate object relational databases over the web. An implicitly generated query retrieves desired sets of properties and entities presented in documents of Extensible Markup Language and Resource Description Framework for further navigation. Container types in Resource Description Framework are mapped by this invention to record and table types in a normalized relational model where URIs locating elements in relational schema components over the web are stored as primary keys/foreign keys in normalized tables. Methods and operators on such web objects are defined as part of user-defined package definitions in object relational schema where object request brokers apply such methods or operators on result sets from relational operations anywhere on the web. This invention uniquely incorporates two distinct stages of SQL computations for a collaborative method of preparation, execution and resolution of an object SQL query over disparate locations of multiple object relational databases on the web.

15 Claims, 24 Drawing Sheets



entity and arrows representing relationships to other resources are added pictorially. An arrow representing relationship R connects to a collection type. Such a creation of resources and their relationships through diagrams of metadata information will be implicitly translated into XML/RDF documents.

In FIG. **18**, a block diagram is shown with thin clients, object relational databases and XML/RDF data/metadata views. XML/RDF documents with metadata definitions along with data sets constitute a view over a database. A part of a single database or union over several databases can be viewed by a specific XML/RDF definition in a document. As the database(s) changes with insert/delete/update, such views can be updated by generating new documents and replacing the old ones. Thin clients consist of windows created from browsers to perform transactions over databases through persistent connections (IIOP). In FIG. **18**, XML/RDF documents can be viewed by browsers to navigate through multiple documents with Uniform Resource Identifiers for finding details of data/metadata definitions. After inspecting data and metadata in one or more XML/RDF documents, a thin client window could be created for transactions over an object relational database for a focused data item(s). Such a transaction could be performed on a data item, data set or text, image contents. Transactions are performed by object relational databases through collaborations and the result set is formatted in XML/RDF to replace existing XML/RDF documents or to create new documents for the client to visualize.

In FIG. **19**, a block diagram shows a logical schema partitioned over multiple physical schemas over the internet. Schema **1** is a component object relational database containing information from union over many XML/RDF document definitions. Schema **2** is another instance of object relational database over the internet similar to schema **1**. Schema **3** is a component object relational system with other relational data. These components communicate with each other through HTT? and IIOP protocols for collaborative query executions. FIG. **19** also shows a block representing a single object SQL query triggered against a unified logical schema. The query gets executed in many component physical schema locations. In FIG. **20**, a block diagram extends the notion in FIG. **19** to show how legacy and other existing relational databases can be seamlessly accessed over the internet by the use of single object SQL triggered against a unified schema. Object Request Brokers and XML/RDF integrators will serve as a middleware for the legacy and existing large installed database systems. Object relational database components over the internet communicate with legacy databases seamlessly through XML/RDF documents and object request brokers. This invention extends the scope of seamless integration of any type of database systems over the internet allowing querying and transactional capabilities through a browser and a thin client window respectively.

I claim:

1. In a network of computers over the internet, a multi-tier client/server system comprising of:

A) Clients with browsers capable of processing documents in Extensible Markup Language (XML) and Resource Description Framework (RDF) carrying data/metadata information for electronic commerce and electronic business where these clients make synchronous communications through Hypertext Transfer Protocol (HTTP) with various internet sites and with object relational database servers at multiple locations consisting of XML/RDF integration software and software compliant with Common Object Request Broker Architecture;

B) Clients with browsers capable of creating and maintaining thin client windows on demand for persistent connectivity through Internet Inter ORB Protocol (IIOP) with remote object relational databases over the web for performing transactions;

C) Relational database servers with schema and business application logic defined in the form of object packages comprising:

a) User-defined packages for providing call specifications for a set of interfaces to embed in Structured Query Language (SQL) queries where these interfaces are implemented as methods in Java classes and are applied to records resulting from execution of relational operations;

b) User-defined packages for specifying operations over attribute values from multiple tables where interface definitions for methods and operators carry parameter definitions along with parameter types defined directly as table names and attribute names for safe execution semantics;

c) User-defined packages specifying interfaces where a parameter type definition maps to another interface existing at a local or remote schema object on the internet;

d) User-defined packages specifying interfaces where parameter type definitions map to tables and attributes existing at a local or remote schema object on the internet;

e) User-defined packages where Uniform Resource Identifiers (URIs) are used to locate elements in remote schema objects on the web and such URIs defining schema objects are stored in data dictionaries or repositories;

f) Method and operator interfaces defined in user-defined packages implemented in Java classes;

D) Executions of application logic in object packages on local or remote schema objects performed by Object Request Broker services in Common Object Request Broker Architecture (CORBA).

2. A system as defined in claim **1** wherein said clients and said object relational database servers communicate through documents in Extensible Markup Language and Resource Description Framework as well as thin client persistent windows for transactions and wherein said multiple relational database servers further communicate among themselves through documents in Extensible Markup Language and Resource Description Framework as well as through persistent connectivity in Internet Inter ORB Protocol (IIOP), switching roles as clients and servers.

3. In a network of computers over the internet, a multi-tier client/server system comprising of:

(a) Multiple related internet sources representing documents in Extensible Markup Language (XML) and Resource Description Framework (RDF) for describing data and metadata for various resources and their interrelationships necessary for electronic commerce and electronic business;

(b) Resource Description Framework (RDF) describing properties and values of properties for resources identified by Uniform Resource Identifiers (URIs) where these URIs are used as primary keys and foreign keys in a normalized relational model of description for these said relationships over resources;

(c) Resource Description Framework (RDF) describing properties and values of properties for resources mapped to tables with foreign key/primary key rela-

tionships in one or more relational databases where foreign key attributes may be named to express same or inverse meaning of corresponding relationship names in RDF description, these names being defined in XML namespaces and schema definitions;

(d) Resource Description Framework (RDF) describing properties and values of properties for resources further comprises:

(1) RDF container types, namely Bag, Sequence and Alternative to represent collection of resources where these container types are mapped to normalized relations in relational model;

(2) RDF Bag container type with one-to-many relationships mapped to two tables in first normal form where one table contains primary key as the URI of the resource under description and the other table contains primary keys as the URIs of the resources in Bag container along with foreign key inherited from first relation;

(3) RDF Bag container type with many-to-many relationships mapped to three tables in second normal form where first table contains primary key as the URI of a resource under description, second table contains primary keys as the URIs of resources in Bag container and third table maps primary keys of first table to primary keys of second table capturing many to many relationships;

(4) RDF Bag container type with more complex nesting of relationships normalized into relations in higher normal forms;

(5) RDF collection type Alternative (Alt) mapped to two relations in one-to-one relationship where the first relation contains records with a two part primary key, first part as the URI of the resource under description and second part a tag to identify a specific alternative, and the second relation contains primary key as URI of a resource in Alternative container along with foreign key being the two part primary key inherited from first relation;

(6) RDF collection type Sequence (Seq) mapped to a single relation with multi-part primary key where first part is the URI of the resource under description and second part is URI of a resource in Sequence container maintaining sequence by sorting over compound primary key;

(e) Data and metadata information stored in a relational database or part of a relational database or union over multiple relational databases mapped to documents in Extensible Markup Language and Resource Description Framework for entities, properties and values of properties where such entities and properties are URIs identifying table names and attribute names respectively.

4. A system as defined in claim 3 wherein said documents in Extensible Markup Language and Resource Description Framework further represent queries in Structured Query Language (SQL) by defining operators and key words of SQL language in Extensible Markup Language namespace schema definitions and wherein operands to such relational operators are either resources represented by Uniform Resource Identifiers or literals.

5. A system as defined in claim 4 wherein said Structured Query Language (SQL) queries embed method interfaces represented in Extensible Markup Language and Resource Description Framework and wherein such method interface names along with all other parameter names within method interfaces are defined in Extensible Markup Language namespace schema definitions.

6. A system as defined in claims 5 wherein said object SQL queries with or without embedded method interfaces are implicitly generated as a result of property based navigation and information requirement specification from multiple related documents in Extensible Markup Language and Resource Description Framework.

7. A system as defined in claim 6 wherein said implicitly generated client queries travel to said object relational databases for collaborative execution of said queries and wherein said object relational database servers further generate documents in Extensible Markup Language and Resource Description Framework for representing results of such said queries for sending back to said clients for further navigation.

8. A system as defined in claim 3 wherein said multiple documents in Extensible Markup Language and Resource Description Framework constitute different views over a unified Database consisting of multiple object relational databases over the web and wherein such views are updated by generating new documents to replace old ones as changes are made to said databases by inserts, deletes and updates.

9. A system as defined in claim 3 wherein said metadata information inside a unified Database over multiple disparate relational databases over the web are possible to be extracted for visualizing entities with various relationships without fetching detailed values of attributes from databases and wherein such entity-relationship diagrams represented in Extensible Markup Language and Resource Description Framework are sent to client locations for visual inspection, manipulation and focused selection of entities or attributes for further detailed visualization, thereby saving critical traversal time of large contents and data values of unnecessary entities and attributes over the internet.

10. A system as defined in claim 9 wherein said entity-relationship diagrams representing metadata information over resources and relationships are visualized through software visual tools for various manipulations and wherein such tools are used to create new resources with relationships to existing entities or collection of entities, forming new web pages by implicit translations of such said entity-relationship diagrams into documents in Extensible markup Language and Resource Description Framework.

11. In a network of computers over the internet, a multi-tier client/server system incorporating distributed business application logic with relational data at disparate locations comprising of:

A) Multiple documents in Extensible Markup Language and Resource Description Framework with data and metadata definitions to view, navigate and trigger implicit queries against a logical schema definition for a single unified Database consisting of multiple disparate relational database instances with tables, foreign key/primary key relationships, user-defined types and business application logic in object packages;

B) Multiple documents in Extensible Markup Language and Resource Description Framework with data and metadata definitions for navigating through a single unified schema containing many instances of component physical schema of relational databases, so as to locate and focus on a data set(s), a text item or any other content(s) and to initiate transaction on such focused item(s) in a separately created thin client window(s) establishing persistent connectivity with any remote physical database instance(s);

C) A single unified virtual Database for navigation, transaction processing and implicit object relational query processing over the web, defined over multiple physical component schema consisting of:

a) Uniform Resource Identifiers (URIs) used for locating remote schema objects;

b) Relations as view caches to maintain relationships over foreign keys inherited from a table in the local component schema and primary keys of a table in a remote schema located by URIs;

c) Relations as view caches to enforce referential integrity across component physical relational databases over the internet;

d) Business logic in the form of object packages implemented in Java classes to incorporate processing logic over a join or other relational operations performed in a local or remote component schema;

e) View cache tables maintaining an attribute(s) which is an user-defined type with an associated object package implemented as Java class at a remote schema location and URIs as attribute values for locating primary keys in that remote schema.

D) Property based specification and navigation through multiple documents in Extensible Markup Language and Resource Description Framework for implicitly generating object SQL queries to represent a multiple resource join in a single logical schema where such a join is performed as multiple partial joins at disparate component physical schema with communications through Internet Inter ORB Protocol and Hypertext Transfer Protocol at each schema location;

E) Property based navigation through multiple documents in Extensible Markup Language and Resource Description Framework for implicitly generating and executing object SQL queries leading to results from many levels of business logic applications on a complex join in a logical schema where such a business logic application on a complex join is executed as cooperative processing of distributed business logic on partial joins in component schema at disparate locations communicating through Object Request Brokers;

F) Object SQL query processing over a single unified database over the internet in two separate stages of processing where stage (1) of processing resolves metadata definitions from disparate schema components to decompose an object SQL query into separate components and stage (2) of processing executes such SQL components in physical databases;

G) Object SQL query executions in a single unified Database over the internet leading to collaborations in stage (1) of computations by: (a) sending proper sets of records to remote schema locations, (b) receiving resulting records from remote schema locations, (c) creating temporary tables in local database for storing record sets received and (d) triggering necessary SQL components to perform join over record sets from local and remote databases.

12. A multi-tier client/server system defined in claim 11 wherein said virtual unified Database over disparate database components comprises two distinct SQL processing for stage (1) and stage (2) of computations and wherein each physical database component over the internet contains separate software layers for collaborative object SQL processing in stage (1) of computation.

13. A system as defined in claim 11 wherein said logical schema for a unified Database partitioned into said physical schema components comprises means of compiling object SQL (Structured Query Language) queries triggered directly or expressed in Resource Description Framework documents and executing such queries against the said logical schema irrespective of locations and distribution of physical schema components by the steps including:

A) Keeping non-detailed metadata (information about relational data) for the logical schema at each component physical schema locations or at a centralized repository so that locations of other component schema are available to said stage (1) computation of SQL processing for all tables, types and packages used in a query;

B) Parsing a Structured Query Language (SQL) query at a physical component schema location by said stage (1) SQL processing layer to resolve definitions for tables, types and packages whatever is locally available and extracting portion of the query for sending it to other locations for preparation by corresponding said stage (1) SQL processing layers at disparate locations wherever the relevant metadata is available;

C) Preparing fully a SQL query at multiple sites by successively parsing with locally available metadata and initializing any object packages used at each component schema location by said stage (1) SQL processing software layers;

D) Beginning execution only after complete preparations at all involved locations;

E) Performing relational operations by said stage (2) SQL processing layers and executing application logic at each component schema location by collaborations at said stage (1) SQL processing layers through Object Request Brokers;

F) Completing any business logic execution at a location depends on successive completion of other executions at other locations when there are dependencies with nested applications and generating result sets in Extensible Markup Language and Resource Description Framework by said stage (1) SQL processing layers for transporting back to client locations.

14. A system as defined in claim 13 wherein said preparation phase and said execution phase of an SQL query further comprises means of including internet locations with centralized legacy databases and other existing data stores with middleware software for processing documents in Extensible Markup Language and Resource Description Framework and also middleware software compliant with Common Object Request Broker Architecture.

15. A system as defined in claim 14 wherein said legacy databases and said existing centralized data stores carrying no object packages for business logic and carrying necessary middleware software for processing documents in Extensible Markup Language and Resource Description Framework further comprises means of supporting SQL queries with embedded business logic against such legacy databases by defining object packages in other component schema locations.

* * * * *

65/7/22 (Item 22 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

0012890625 *Drawing available*
WPI Acc no: 2002-750095/200281
Related WPI Acc No: 2001-570736; 2002-558153
XRPX Acc No: N2002-590767

Computer readable medium has instructions for mapping identifiers of parent and child relationship in semi-structured data to table identifier in structured data

Patent Assignee: CHRISTIANSON D B (CHRI-I); DRAPER D L (DRAP-I); KOMMISSARCHIK K L (KOMM-I)
Inventor: CHRISTIANSON D B; DRAPER D L; KOMMISSARCHIK K L

Patent Family (1 patents, 1 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
US 20020120630	A1	20020829	US 2000517131	A	20000302	200281	B
			US 2000517468	A	20000302		
			US 2000718228	A	20001121		
			WO 2001US6755	A	20010302		
			US 2001921010	A	20010801		

NIMBLE TECHNOLOGY

Priority Applications (no., kind, date): WO 2001US6755 A 20010302; US 2000718228 A 20001121; US 2000517468 A 20000302; US 2000517131 A 20000302; US 2001921010 A 20010801

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
US 20020120630	A1	EN	26	15	C-I-P of application	US 2000517131
					C-I-P of application	US 2000517468
					C-I-P of application	US 2000718228
					C-I-P of application	WO 2001US6755

Alerting Abstract US A1

NOVELTY - The computer readable medium has an instruction for generating a structured organization (52) including tables to store the collection of semi-structured **data**. Another instruction **maps** the identifiers of the parent and child **relationships** in the semi-structured **data** to a **table identifier** in a structured **data** that corresponds to the parent and child **relationship**.
DESCRIPTION - INDEPENDENT CLAIMS are included for the following:

1. **Data structure** generation method;
2. Structured organization generation method;
3. Structured organization generating apparatus;
4. Structured organization generating **program**;

5. Semi-structured **data** organization generating method;
6. Semi-structured **data** organization generating apparatus;
7. Semi-structured **data** organization **generating program**;
8. Semi-structured **queries** pre-processing method;
9. Semi-structured queries pre-processing apparatus; and
10. Semi-structured queries pre-processing **program**.

USE - For **mapping** semi-structured **data** such as **extensible mark-up language (XML)** encoded **data** to structured **data** such as those used for **database**, spread sheet, address book, etc.
ADVANTAGE - Enables vast volume of legacy structured **data** to be employed in modern information pages by using semi-structured **data**, thereby enabling speed up of fulfillment of requested information pages that have to be dynamically completed with semi-structured **data** retrieved in real-time.
DESCRIPTION OF DRAWINGS - The figure shows an overview of the **mapping** procedure.
52 Structured organization

Title Terms /Index Terms/Additional Words: COMPUTER; READ; MEDIUM; INSTRUCTION; **MAP**; IDENTIFY; PARENT; CHILD; RELATED; SEMI; STRUCTURE; **DATA**; TABLE

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-007/00			Main		"Version 7"

US Classification, Issued: 707103R00

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): **T01-J05B1**; **T01-J05B2B**; **T01-J11C1**; **T01-S01B**; **T01-S03**

Original Publication Data by Authority

United States

Publication No. US 20020120630 A1 (Update 200281 B)
Publication Date: 20020829
Method and apparatus for storing semi-structured data in a structured manner
Assignee: Christianson, David B., Seattle, WA, US (CHRI-I)
Draper, Denise L., Seattle, WA, US (DRAP-I)
Kommissarchik, Konstantin L., Brier, WA, US (KOMM-I)
Inventor: Christianson, David B., Seattle, WA, US
Draper, Denise L., Seattle, WA, US
Kommissarchik, Konstantin L., Brier, WA, US
Agent: PERKINS COIE LLP, PATENT-SEA, P.O. BOX 1247, SEATTLE, WA, US

Language: EN (26 pages, 15 drawings)
Application: US 2000517131 A 20000302 (C-I-P of application)
US 2000517468 A 20000302 (C-I-P of application)
US 2000718228 A 20001121 (C-I-P of application)
WO 2001US6755 A 20010302 (C-I-P of application)
US 2001921010 A 20010801 (Local application)
Original IPC: G06F-7/00(A)
Current IPC: G06F-7/00(A)
Original US Class (main): 707103.R
Original Abstract: In response to the provision of a collection of semi-structured data (or its schema), a mapper generates a structured organization to store the collection of semi-structured data. The mapper further collaterally generates a description of how the semi-structured data are stored under the structured organization. In one embodiment, normalized relational tables are used for the structured organization. In alternate embodiments, non-normalized relational tables or other structured organization may also be used. The reverse, that is, the creation of a semistructured data organization for a collection of structured data may also be performed.
Claim: What is claimed is:

1. 1. A computer-readable medium containing a data structure for mapping between semi-structured data and structured data, the data structure comprising:
 - for each parent and child relationship in the semi-structured data, a mapping from a parent identifier and a child identifier of the parent and child relationship to a table identifier of a table in the structured data that corresponds to the parent and child relationship.

US 20020120630A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2002/0120630 A1

Christianson et al. (43) Pub. Date: Aug. 29, 2002

(54) METHOD AND APPARATUS FOR STORING SEMI-STRUCTURED DATA IN A STRUCTURED MANNER

(76) Inventors: David B. Christianson, Seattle, WA (US); Denise L. Draper, Seattle, WA (US); Konstantin L. Kommissarchik, Brier, WA (US)

Correspondence Address:
PERKINS COIE LLP
PATENT-SEA
P.O. BOX 1247
SEATTLE, WA 98111-1247 (US)

(21) Appl. No.: 09/921,010

(22) Filed: Aug. 1, 2001

Related U.S. Application Data

(63) Continuation-in-part of application No. 09/517,131, filed on Mar. 2, 2000, which is a continuation-in-part of application No. PCT/US01/06755, filed on Mar. 2, 2001, which is a continuation-in-part of application No. 09/718,228, filed on Nov. 21, 2000, which is a continuation-in-part of application No. 09/517,468, filed on Mar. 2, 2000.

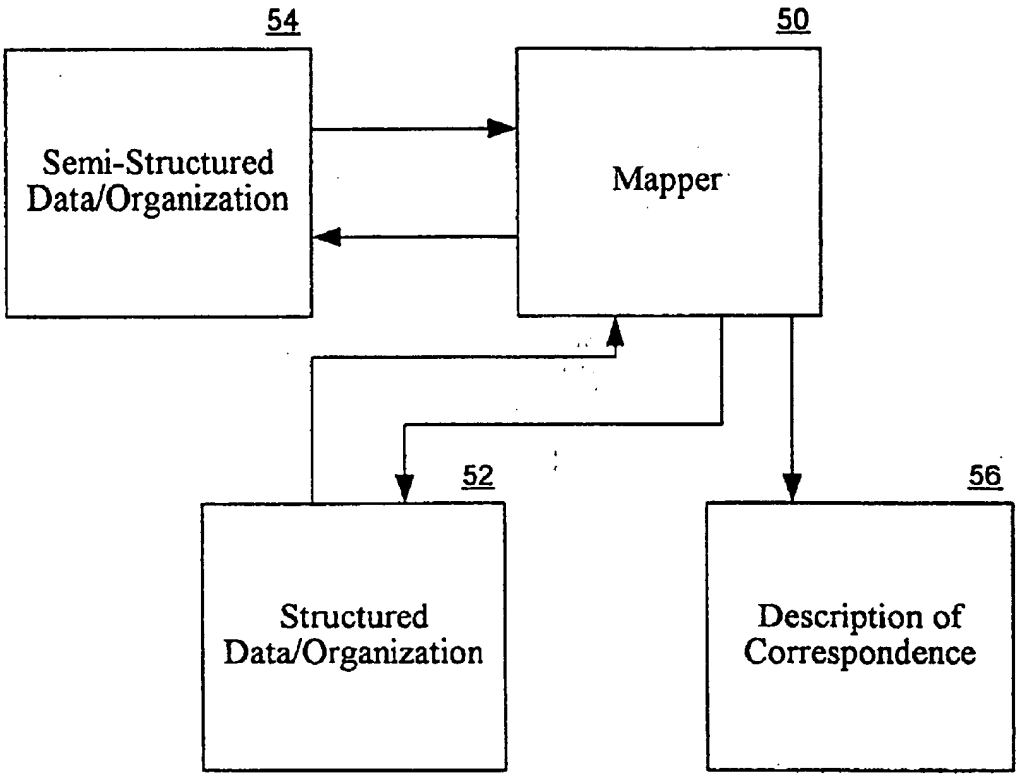
Publication Classification

(51) Int. Cl.[7] G06F 7/00

(52) U.S. Cl. 707/103 R

(57) ABSTRACT

In response to the provision of a collection of semi-structured data (or its schema), a mapper generates a structured organization to store the collection of semi-structured data. The mapper further collaterally generates a description of how the semi-structured data are stored under the structured organization. In one embodiment, normalized relational tables are used for the structured organization. In alternate embodiments, non-normalized relational tables or other structured organization may also be used. The reverse, that is, the creation of a semistructured data organization for a collection of structured data may also be performed.



scope of the appended claims. The description is thus to be regarded as illustrative instead of restrictive on the present invention.

What is claimed is:

1. A computer-readable medium containing a data structure for mapping between semi-structured data and structured data, the data structure comprising:

for each parent and child relationship in the semi-structured data, a mapping from a parent identifier and a child identifier of the parent and child relationship to a table identifier of a table in the structured data that corresponds to the parent and child relationship.

2. The computer-readable medium of claim 1 wherein each mapping further includes a parent key identifier and a child key identifier of the table that correspond to the parent identifier and child identifier of the mapping.

3. The computer-readable medium of claim 2 wherein each mapping further includes an indicator of whether a child of the parent and child relationship is an attribute or an element.

4. The computer-readable medium of claims **1-3** wherein the data structure supports recursive definition of elements of the semi-structured data.

5. The computer-readable medium of claims **1-4** wherein the semi-structured data is in XML format.

6. The computer-readable medium of claims **1-5** wherein the structured data is in a relational format.

7. The computer-readable medium of claims **1-6** wherein the data structure includes for each child of a parent and child relationship that is also not a parent, a mapping from the child identifier of that child to a table identifier of a table in the structured data.

8. The computer-readable medium of claim 7 wherein the mapping from the child identifier of that child also maps to a child key identifier in the table.

9. The computer-readable medium of claims **7-8** wherein the mapping from the child identifier of that child also maps to an indicator as to whether the child is an attribute or an element of its parent.

10. The computer-readable medium of claims **7-9** wherein the mapping from the child identifier of that child also maps to a child value identifier in the table.

11. A method for generating a data structure for mapping between semi-structured data and structured data, the method comprising:

identifying parent and child relationships in the semi-structured data; for each identified parent and child relationship,

identifying a table in the structured data that corresponds to the identified parent and child relationship; and

creating a mapping between the identified parent and child relationship and the identified table.

12. The method of claim 11 wherein the mapping maps a parent identifier and child identifier of the parent and child relationship to a table identifier of the identified table.

13. The method of claim 12 wherein each mapping further includes a parent key identifier and a child key identifier of the table that correspond to the parent identifier and child identifier of the mapping.

14. The method of claim 13 wherein each mapping further includes an indicator of whether a child of the parent and child relationship is an attribute or an element.

15. The method of claims **12-14** wherein the method supports recursive definition of elements of the semi-structured data.

16. The method of claims **12-15** wherein the semi-structured data is in XML format.

17. The method of claims **12-16** wherein the structured data is in a relational format.

18. The method of claims **12-17** including for each child of a parent and child relationship that is also not a parent, creating a mapping from the child identifier of that child to a table identifier of a table in the structured data.

19. The method of claim 18 wherein the mapping from the child identifier of that child also maps to a child key identifier in the table.

20. The method of claims **18-19** wherein the mapping from the child identifier of that child also maps to an indicator as to whether the child is an attribute or an element of its parent.

21. The method of claims **18-20** wherein the mapping from the child identifier of that child also maps to a child value identifier in the table.

22. A method comprising:

generating a structured organization to store a collection of semi-structured data; and

collaterally generating a description of how the semi-structured data is stored under the structured organization.

23. The method of claim 22, wherein said collateral generation of a description comprises generating descriptive entries for a description table, with each descriptive entry identifying a first and a second storage column of a storage table for a parent and a child node of a parent and child node pair of the semi-structured data.

24. The method of claim 23, wherein said collateral generation of a description further comprises generating for each of a selected one or ones of said descriptive entries a context qualifier for at least a selected one of the parent and the child node.

25. The method of claim 22, wherein said generation of a structured organization comprises creating relational storage tables with rows and columns in a normalized manner.

26. An apparatus comprising:

a storage medium having stored therein a plurality of programming instructions designed to generate a structured organization to store a collection of semi-structured data, when executed, and to collaterally generate a description of how the semi-structured data are stored under the structured organization; and a processor coupled to the storage medium to execute the programming instructions.

27. The apparatus of claim 26, wherein the programming instructions are designed to generate the description by generating a description table having a plurality of descriptive entries, with each descriptive entry identifying a first and a second storage column of a storage table of the structured organization for a parent and a child node of a parent and child node pair of the semi-structured data.

28. The apparatus of claim 27, wherein said programming instructions are further designed to generate for each of a

selected one or ones of said descriptive entries a context qualifier for at least a selected one of the parent and the child node.

29. The apparatus of claim 26, wherein said programming instructions are designed to generate a structured organization by creating relational storage tables with rows and columns in a normalized manner.

30. The apparatus of claim 26, wherein said programming instructions are a subset of a larger collection of programming instructions implementing a selected one of a database manager and a development toolkit that facilitates development of applications that access databases.

31. An article of manufacture comprising:

a recordable medium having recorded thereon a plurality of programming instructions for use to program a system having a processor for executing programming instructions, to enable the system to generate a structured organization to store a collection of semi-structured data, and to collaterally generate a description of how the semi-structured data are stored under the structured organization.

32. The article of claim 31, wherein said programming instructions enable the system to collaterally generating a description by generating a description table having a plurality of descriptive entries, with each descriptive entry identifying a first and a second storage column of a storage table for a parent and a child node of a parent and child node pair of the semi-structured data.

33. The article of claim 32, wherein said programming instructions further enable the system to generate for each of a selected one or ones of said descriptive entries a context qualifier for at least a selected one of the parent and the child node.

34. The article of claim 31, wherein said programming instructions enable the system to generate a structured organization by creating relational storage tables with rows and columns in a normalized manner.

35. The article of claim 31, wherein said programming instructions are a subset of a larger collection of programming instructions implementing a selected one of a database manager and a development toolkit that facilitates development of applications that access databases.

36. A method comprising:

generating a semi-structured data organization for a collection of structured data; and

collaterally generating a description of correspondence between the generated semi-structured data organization and the collection of structured data.

37. The method of claim 36, wherein said collateral generation of a description comprises generating descriptive entries for a description table, with each descriptive entry identifying a first and a second storage column of a storage table of the structured data for a parent and a child node of a parent and child node pair of the generated semi-structured data organization.

38. The method of claim 37, wherein said collateral generation of a description further comprises generating for each of a selected one or ones of said descriptive entries a context qualifier for at least a selected one of the parent and the child node.

39. The method of claim 37, wherein said generation of a semi-structured data organization comprises creating an extensible mark-up language (XML) data structure for a collection of relational tables.

40. An apparatus comprising: a storage medium having stored therein a plurality of programming instructions designed to generate a semi-structured data organization for a collection of structured data, when executed, and to collaterally generate a description of correspondence between the semi-structured data organization and the structured data; and a processor coupled to the storage medium to execute the programming instructions.

41. The apparatus of claim 40, wherein the programming instructions are designed to generate the description by generating a description table having a plurality of descriptive entries, with each descriptive entry identifying a first and a second storage column of a storage table of the structured data for a parent and a child node of a parent and child node pair of the generated semi-structured data organization.

42. The apparatus of claim 41, wherein said programming instructions are further designed to generate for each of a selected one or ones of said descriptive entries a context qualifier for at least a selected one of the parent and the child node.

43. The apparatus of claim 40, wherein said programming instructions are designed to create an extensible mark-up language (XML) data structure for a collection of relational tables.

44. The apparatus of claim 40, wherein said programming instructions are a subset of a larger collection of programming instructions implementing a selected one of a database manager and a development toolkit that facilitates development of applications that access databases.

45. An article of manufacture comprising:

a recordable medium having recorded thereon a plurality of programming instructions for use to program a system having a processor for executing programming instructions, to enable the system to generate a semi-structured data organization for a collection of structured data, and to collaterally generate a description of correspondence between the semi-structured data organization and the structured data.

46. The article of claim 45, wherein said programming instructions enable the system to collaterally generating a description by generating a description table having a plurality of descriptive entries, with each descriptive entry identifying a first and a second storage column of a storage table of the structured data for a parent and a child node of a parent and child node pair of the generated semi-structured data organization.

47. The article of claim 46, wherein said programming instructions further enable the system to generate for each of a selected one or ones of said descriptive entries a context qualifier for at least a selected one of the parent and the child node.

48. The article of claim 45, wherein said programming instructions enable the system to create an extensible mark-up language (XML) data structure for a collection of relational tables.

49. The article of claim 45, wherein said programming instructions are a subset of a larger collection of programming instructions implementing a selected one of a database manager and a development toolkit that facilitates development of applications that access databases.

50. A method comprising:

pre-processing semi-structured queries in an information page employing semi-structured data to identify the data to be selectively retrieved from the semi-structured data, and the storage locations of the data in a collection of structured data comprising the semi-structured data, referencing a description of how the semi-structured data are stored in the collection of structured data;

generating equivalent structured queries to retrieve the data from the collection of structured data; and

replacing the semi-structured queries with the equivalent structured queries.

51. The method of claim 50, wherein said pre-processing of semi-structured queries is also performed to identify join conditions for joining relational storage tables of said collection of structured data to retrieve the data.

52. The method of claim 50, wherein said pre-processing of semi-structured queries comprises pre-processing matching templates of said semi-structured queries.

53. The method of claim 52, wherein said pre-processing matching templates comprises pre-processing of data extraction commands within each of the matching templates.

54. The method of claim 50, wherein said generation of equivalent structured queries comprises generating structured queries, and control structures controlling repeated execution of the generated structured queries.

55. An apparatus comprising:

a storage medium having stored therein a plurality of programming instructions designed

to pre-process semi-structured queries in an information page employing semi-structured data to identify the data to be selectively retrieved from the semi-structured data, and the storage locations of the data in a collection of structured data comprising the semi-structured data, referencing a description of how the semi-structured data are stored in the collection of structured data,

to generate equivalent structured queries to retrieve the data from the collection of structured data, and

to replace the semi-structured queries with the equivalent structured queries; and

a processor coupled to the storage medium to execute the programming instructions.

56. The apparatus of claim 55, wherein said programming instructions are also designed to pre-process the semi-structured queries to identify join conditions for joining relational storage tables of said collection of structured data to retrieve the data.

57. The apparatus of claim 55, wherein said programming instructions are designed to pre-process matching templates of said semi-structured queries.

58. The apparatus of claim 57, wherein said programming instructions are further designed to pre-process data extraction commands within each of the matching templates.

59. The apparatus of claim 55, wherein said programming instructions are designed to generate structured queries, and control structures controlling repeated execution of the generated structured queries.

60. The apparatus of claim 55, wherein said programming instructions are a subset of a larger collection of programming instructions implementing a selected one of a database manager and a development toolkit that facilitates development of applications that access data bases.

61. An article of manufacture comprising:

a recordable medium having recorded thereon a plurality of programming instructions for use to program a system having a processor for executing programming instructions, to enable the system

to pre-process semi-structured queries in an information page employing semi-structured data to identify the data to be selectively retrieved from the semi-structured data, and the storage locations of the data in a collection of structured data comprising the semi-structured data, referencing a description of how the semi-structured data are stored in the collection of structured data,

to generate equivalent structured queries to retrieve the data from the collection of structured data, and

to replace the semi-structured queries with the equivalent structured queries.

62. The article of claim 61, wherein said programming instructions are also designed to enable the system to pre-process the semi-structured queries to identify join conditions for joining relational storage tables of said collection of structured data to retrieve the data.

63. The article of claim 61, wherein said programming instructions are designed to enable the system to pre-process matching templates of said semi-structured queries.

64. The article of claim 63, wherein said programming instructions are further designed to enable the system to pre-process data extraction commands within each of the matching templates.

65. The article of claim 61, wherein said programming instructions are designed to enable the system to generate structured queries, and control structures controlling repeated execution of the generated structured queries.

66. The article of claim 61, wherein said programming instructions are a subset of a larger collection of programming instructions implementing a selected one of a database manager and a development toolkit that facilitates development of applications that access data bases.

67. A method comprising:

storing pre-processed information pages formed with a language that provides for semi-structured queries, and employing semi-structured data, where the semi-structured queries have been pre-replaced with equivalent structured queries to retrieve the data to be retrieved by the semi-structured queries from the semi-structured data, from a collection of structured data comprising the semi-structured data instead;

receiving a request for a selected one of the information pages;

dynamically completing the information page, executing the replacement structured queries; and

returning the dynamically completed information page.

68. The method of claim 67, wherein the method further comprises

pre-processing the semi-structured queries of each information page employing semi-structured data to identify the data to be selectively retrieved from the semi-

structured data, and the storage locations of the data in the collection of structured data comprising the semi-structured data, referencing a description of how the semi-structured data are stored in the collection of structured data

generating equivalent structured queries to retrieve the data from the collection of structured data; and

replacing the semi-structured queries with the equivalent structured queries.

69. The method of claim 68, wherein said pre-processing of semi-structured queries is also performed to identify join conditions for joining relational storage tables of the collection of structured data to retrieve the data.

70. The method of claim 68, wherein said generation of equivalent structured queries comprises generating structured queries, and control structures controlling repeated execution of the generated structured queries.

71. An apparatus comprising:

a network interface coupling the apparatus to a data network;

a storage medium having stored therein pre-processed information pages formed with a language that provides for semi-structured queries, and employing semi-structured data, where the semi-structured queries have been pre-replaced with equivalent structured queries to retrieve the data to be retrieved by the semi-structured queries from the semi-structured data, from a collection of structured data comprising the semi-structured data instead; and

a processor coupled to the network interface and the storage medium to receive a request from a client for a selected one of the information pages from the data network, and in response, dynamically completing the information page, executing the replacement structured queries, and returning the dynamically completed information page to the client through the data network.

72. The apparatus of claim 71, wherein the apparatus is a web server, the language is extensible script language (XSL), the information pages are XSL documents, and the semi-structured data are extensible mark-up language (XML) data.

73. The apparatus of claim 72, wherein the structured data are relational data and the structured queries are formed using the structured query language (SQL).

* * * * *

65/7/13 (Item 13 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

0013350568 *Drawing available*
WPI Acc no: 2003-438415/200341
XRPX Acc No: N2003-349736
Multimedia data storage method involves storing XML formatted logical annotations obtained from meta data in media file and auxiliary data, along with media file in database
Patent Assignee: CARRER M (CARR-I); LIN P (LINP-I); SRIVASTAVA A (SRIV-I)
Inventor: CARRER M; LIN P; SRIVASTAVA A

Patent Family (1 patents, 1 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
US 6549922	B1	20030415	US 1999410781	A	19991001	200341	B

ORACLE

Priority Applications (no., kind, date): US 1999410781 A 19991001

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
US 6549922	B1	EN	9	3		

Alerting Abstract US B1
NOVELTY - The **meta data** stored in a particular media **file** and information characterizing the **file** content are acquired. Auxiliary **data** characterizing the **file** are acquired from other external sources. The acquired **data** are combined to form a set of logical annotations which is formatted into **XML data** . The media **file** and formatted logical annotations are stored in a **database** .
DESCRIPTION - An INDEPENDENT CLAIM is also included for multimedia **data** collecting and storing apparatus.
USE - For collecting and storing digital multimedia **data**.
ADVANTAGE - The **data** stored in digital media is readily managed, indexed and searched by the **application program**, without using special programming by utilizing the **database** including the original media **file** and the formatted logical annotations.
DESCRIPTION OF DRAWINGS - The figure shows the block diagram of the apparatus for collecting and storing **meta data** on a **database**.

Title Terms /Index Terms/Additional Words: **DATA**; STORAGE; METHOD; LOGIC; OBTAIN; META; MEDIUM; **FILE**; AUXILIARY; **DATABASE**

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-012/00			Main		"Version 7"

G06F-015/00			Secondary		"Version 7"

US Classification, Issued: 707205000, 707512000

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): **T01-F05E**; **T01-J05B1**; **T01-J05B2B**

Original Publication Data by Authority

United States

Publication No. US 6549922 B1 (Update 200341 B)
Publication Date: 20030415
System for collecting, transforming and managing media metadata
Assignee: Srivastava, Alok, One Oracle Dr., Nashua, NH 03060, US (SRIV-I)
Carrer, Marco, One Oracle Dr., Nashua, NH 03062, US (CARR-I)
Lin, Paul, One Oracle Dr., Nashua, NH 03060, US (LINP-I)
Inventor: Srivastava, Alok, One Oracle Dr., Nashua, NH 03060, US
Carrer, Marco, One Oracle Dr., Nashua, NH 03062, US
Lin, Paul, One Oracle Dr., Nashua, NH 03060, US
Language: EN (9 pages, 3 drawings)
Application: US 1999410781 A 19991001 (Local application)
Original IPC: G06F-12/00(A) G06F-15/00(B)
Current IPC: G06F-12/00(A) G06F-15/00(B)
Original US Class (main): 707205
Original US Class (secondary): 707512
Original Abstract: An extensible framework for the automatic extraction and transformation of metadata into logical annotations. Metadata imbedded within a media file is extracted by a type-specific parsing module which is loaded and executed based on the mimetype of the media file being described. A content processor extracts information, typically in the form of time-based samples, from the media content. An auxiliary processing step is performed to collect additional metadata describing the media file from sources external to the file. All of the metadata thus collected is combined into a set of logical annotations, which may be supplemented by summary data generated from the metadata already collected. The annotations are then formatted into a standardized form, preferably XML, which is then mapped into a database schema. The database object also stores the source XML data as well as the original media file in addition to the annotation metadata. The system provides unified metadata repositories, which can then be used for indexing and searching.
Claim: What is claimed is:

1. 1. The method for storing multimedia data comprising, in combination, the steps of:
 - determining the particular media type of a file of data to be stored,
 - selecting and executing a metadata extracting program to extract metadata stored in said file of data in accordance with the file format specifications for said particular media type,
 - selecting and executing a sample extraction program for analyzing the content of said file of data to generate

additional information characterizing the content of said file,

- selecting and executing a program for acquiring auxiliary data characterizing said file of data from one or more sources other than said file of data,
- combining said metadata, said additional information and said auxiliary data for form a set of logical annotations to said file of data,
- formatting said logical annotations into a standard data representation, and
- storing the combination of said file of data and logical annotations in a database.

US006549922B1

(12) **United States Patent**
Srivastava et al.

(10) **Patent No.: US 6,549,922 B1**
(45) **Date of Patent: Apr. 15, 2003**

(54) **SYSTEM FOR COLLECTING, TRANSFORMING AND MANAGING MEDIA METADATA**

(76) Inventors: **Alok Srivastava**, One Oracle Dr., Nashua, NH (US) 03060; **Marco Carrer**, One Oracle Dr., Nashua, NH (US) 03062; **Paul Lin**, One Oracle Dr., Nashua, NH (US) 03060

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **09/410,781**

(22) Filed: **Oct. 1, 1999**

(51) **Int. Cl.**[7] **G06F 12/00**; G06F 15/00
(52) **U.S. Cl.** **707/205**; 707/512
(58) **Field of Search** 707/2–7, 100–101, 707/500–513, 200, 205, 522–524; 345/700–706

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,920,856 A * 7/1999 Syeda-Mahmood 707/3
5,970,490 A * 10/1999 Morgenstern 707/10
6,170,011 B1 * 1/2001 Macleod Beck et al. ... 709/224
6,311,194 B1 * 10/2001 Sheth et al. 707/505

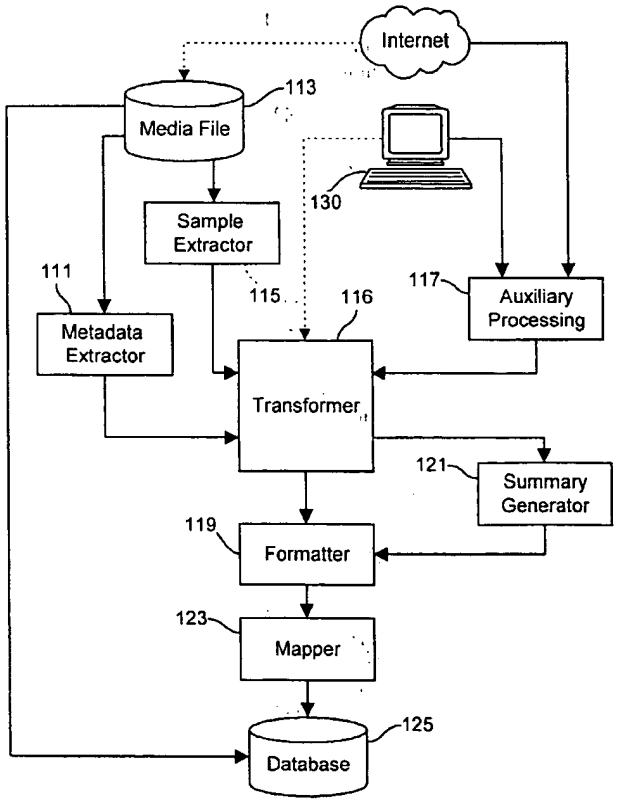
* cited by examiner

Primary Examiner—Alford W. Kindred

(57) **ABSTRACT**

An extensible framework for the automatic extraction and transformation of metadata into logical annotations. Metadata imbedded within a media file is extracted by a type-specific parsing module which is loaded and executed based on the mimetype of the media file being described. A content processor extracts information, typically in the form of time-based samples, from the media content. An auxiliary processing step is performed to collect additional metadata describing the media file from sources external to the file. All of the metadata thus collected is combined into a set of logical annotations, which may be supplemented by summary data generated from the metadata already collected. The annotations are then formatted into a standardized form, preferably XML, which is then mapped into a database schema. The database object also stores the source XML data as well as the original media file in addition to the annotation metadata. The system provides unified metadata repositories, which can then be used for indexing and searching.

10 Claims, 3 Drawing Sheets



It is to be understood that the specific embodiment of the invention which has been described is merely illustrative of one application of the principles of the invention. Numerous modifications may be made to the system described without departing from the true spirit and scope of the invention.

What is claimed is:

1. The method for storing multimedia data comprising, in combination, the steps of:

determining the particular media type of a file of data to be stored,

selecting and executing a metadata extracting program to extract metadata stored in said file of data in accordance with the file format specifications for said particular media type,

selecting and executing a sample extraction program for analyzing the content of said file of data to generate additional information characterizing the content of said file,

selecting and executing a program for acquiring auxiliary data characterizing said file of data from one or more sources other than said file of data,

combining said metadata, said additional information and said auxiliary data for form a set of logical annotations to said file of data,

formatting said logical annotations into a standard data representation, and

storing the combination of said file of data and logical annotations in a database.

2. The method set forth in claim 1 wherein said additional information generated by said sample extraction program includes time-based samples of said file of data.

3. The method set forth in claim 1 wherein one of said sources of auxiliary data include data provide by user characterizing said file of data.

4. The method set forth in claim 1 wherein said standard data representation is the eXtended Markup Language.

5. The method set forth in claim 1 including the further step of processing said logical annotations to produce summary data which is combined with said logical annotations and formatted into said standard data representation.

6. Apparatus for collecting and storing metadata describing a plurality of media data files of different types having different format specifications comprising, in combination,

means for extracting imbedded metadata from each of said media data files,

means for processing the content of each of said media data files to produce information representing said content,

means for accepting external data describing at least selected ones of said files from sources other than said selected ones of said data files,

means for combining said imbedded data, said information representing said content, and said external data to form logical annotation data associated with each of said media data files, and

means for storing said logical annotation data in a standard data format.

7. Apparatus as set forth in claim 6 wherein said information representing said content includes one or more time-based samples of the content of at least some of said media files.

8. Apparatus as set forth in claim 7 further including means for processing said logical annotation data to produce summary data which is stored with said logical annotation data in said standard data format.

9. Apparatus as set forth in claim 6, wherein said standard data format is the eXtensible Markup Language.

10. Apparatus as set forth in claim 6, wherein said means for storing said logical annotation data in a standard data format includes means for mapping at least a portion of said annotation data into a database schema and uploading said portion of said annotation data into a database in accordance with said schema.

* * * * *

65/7/12 (Item 12 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

0013624336 *Drawing available*
WPI Acc no: 2003-719852/200368
Related WPI Acc No: 2004-649379
XRPX Acc No: N2003-575414
Relational data conversion for distributed object application, involves merging tuple stream with construction portion of executable query to generate structured document capable of defining nesting depth
Patent Assignee: AT & T CORP (AMTT)
Inventor: FERNANDEZ M F; SUCIU D; TAN W

Patent Family (1 patents, 1 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
US 6604100	B1	20030805	US 2000181400	P	20000209	200368	B
			US 2001778749	A	20010208		

Priority Applications (no., kind, date): US 2000181400 P 20000209; US 2001778749 A 20010208

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
US 6604100	B1	EN	17	2	Related to Provisional	US 2000181400

Alerting Abstract US B1
NOVELTY - An **executable query** formed by determining the composition of view and user queries, is partitioned into **data** extraction and construction portions. The **data** extraction portion is transmitted to a **relational database** based on which a tuple stream is received. The tuple stream and construction portion are merged to generate a structured document capable of defining arbitrary nesting depth.
DESCRIPTION - INDEPENDENT CLAIMS are also included for the following:

1. computer readable medium storing relational **data** conversion **program**;
2. **computer** system; and
3. composed **query generation** method.

USE - For converting relational **data** into **extensible markup language** (**XML**) for distributed object **application** and inter enterprise **applications** on Internet for healthcare and telecommunication groups.
ADVANTAGE - Provides dynamic and efficient tool for viewing and querying relational **data** converted into **extensible markup language** and reduces the number of queries.
DESCRIPTION OF DRAWINGS - The figure shows the schematic view of relational **data**.
100 silk **route**
104 translator

106 **XML** generator module
110 **relational database** server
120 **application**
130 Internet

Title Terms /Index Terms/Additional Words: RELATED; **DATA**; CONVERT; DISTRIBUTE; OBJECT; APPLY; MERGE; STREAM; CONSTRUCTION; PORTION; EXECUTE; QUERY; GENERATE; STRUCTURE; DOCUMENT; CAPABLE; DEFINE; NEST; DEPTH

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-017/30			Main		"Version 7"

US Classification, Issued: 707003000, 707513000, 707001000

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): **T01-N03B2**; **T01-S03**

Original Publication Data by Authority

United States

Publication No. US 6604100 B1 (Update 200368 B)
Publication Date: 20030805
Method for converting relational data into a structured document
Assignee: AT&T Corp., New York, NY, US (AMTT)
Inventor: Fernandez, Maria F., Madison, NJ, US
Suciu, Dan, Seattle, WA, US
Tan, Wang-Chiew, Philadelphia, PA, US
Language: EN (17 pages, 2 drawings)
Application: US 2000181400 P 20000209 (Related to Provisional)
US 2001778749 A 20010208 (Local application)
Original IPC: G06F-17/30(A)
Current IPC: G06F-17/30(A)
Original US Class (main): 7073
Original US Class (secondary): 707513 7071
Original Abstract: A method for converting relational data to XML (Extensible Markup Language) is provided. The method, sometimes referred to as SilkRoute, provides a general, dynamic and efficient tool for viewing and querying relational data in XML. SilkRoute can express mappings of relational data in XML that conforms to arbitrary public document type definitions. Also, SilkRoute can materialize the fragment of an XML view needed by an application and it can fully exploit the query engine of a relational database management system whenever data items in an

XML view need to be materialized.
Claim: What is claimed:

1. 1. A method for converting relational data from a relational database into a structured document comprising the steps of:
 - storing a view query that defines a structured document view of the relational database, a structure of the view query being independent of a structure of data in the relational database;
 - receiving a user query against the structured document view;
 - forming an executable query by determining a composition of the view query and the user query;
 - partitioning the executable query into a data extraction portion and a construction portion;
 - transmitting the data extraction portion to the relational database;
 - receiving at least one tuple stream from the relational database according to the data extraction portion; and
 - merging the at least one tuple stream and the construction portion to generate a
 - structured document, wherein the structured document view is capable of defining a document of arbitrary nesting depth.

US006604100B1

(12) **United States Patent**
Fernandez et al.

(10) **Patent No.: US 6,604,100 B1**
(45) **Date of Patent: Aug. 5, 2003**

(54) **METHOD FOR CONVERTING RELATIONAL DATA INTO A STRUCTURED DOCUMENT**

(75) Inventors: **Maria F. Fernandez**, Madison, NJ (US); **Dan Suciu**, Seattle, WA (US); **Wang-Chiew Tan**, Philadelphia, PA (US)

(73) Assignee: **AT&T Corp.**, New York, NY (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 212 days.

(21) Appl. No.: **09/778,749**

(22) Filed: **Feb. 8, 2001**

Related U.S. Application Data

(60) Provisional application No. 60/181,400, filed on Feb. 9, 2000.

(51) **Int. Cl.**[7] **G06F 17/30**
(52) **U.S. Cl.** **707/3**; 707/513; 707/1
(58) **Field of Search** 707/1, 2, 3, 4, 707/10, 104.1, 513

(56) **References Cited**

U.S. PATENT DOCUMENTS

6,263,332	B1 *	7/2001	Nasr et al.	707/513
6,343,287	B1 *	1/2002	Kumar et al.	707/4
6,366,934	B1 *	4/2002	Cheng et al.	707/513

OTHER PUBLICATIONS

Jayavel Shanmugasundaram, H. Gang, Kristin Tufte, Chun Zhang, David DeWitt, and Jeffrey F. Naughton. Relational Databases for Querying XML Documents: Limitations and Opportunities. In Proceedings of the Conference on Very Large Data Bases, 1999.*

A. Deutsch, M. F. Fernandez, and D. Suciu. Storing semi-structured data with STORED. In SIGMOD '99, pp. 431–442, 1999.*

Geo Wiederhold, "Mediators in the Architecture of Future Information System," IEEE Computer, Mar. 1992, pp. 38–49.

Mary Fernandez et al., "Efficient Evaluation of XML Middle-ware Queries" Nov. 27, 2000.

Mary Fernandez et al., "SilkRoute: Trading between Relation and XML", Jun. 16, 2000.

I. Manolescu, D. Florescu, D. Kossmann, F. Xhumari, D. Olteanu, "Agora: Living with XML and Relational", in VLDB 2000, pp 623–626.

J. Shanmugasundaram, J. Kiernan, E. Shekita, C. Fan, J. Funderburk, "Querying XML views of relational data", in VLDB 2001.

J. Shanmugasundaram, E. Shekita, R. Barr, M. Carey, B. Lindsay, H. Pirahesh, B. Reinwald, "Efficiently Publishing Relational Data as XML Documents", VLDB Journal (to appear).

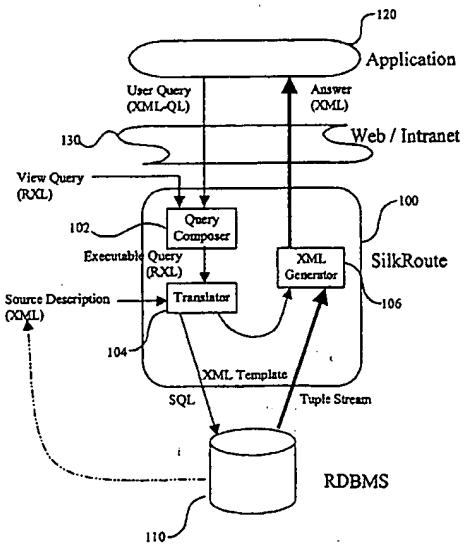
* cited by examiner

Primary Examiner—Diane D. Mizrahi
Assistant Examiner—Yicun Wu

(57) **ABSTRACT**

A method for converting relational data to XML (Extensible Markup Language) is provided. The method, sometimes referred to as SilkRoute, provides a general, dynamic and efficient tool for viewing and querying relational data in XML. SilkRoute can express mappings of relational data in XML that conforms to arbitrary public document type definitions. Also, SilkRoute can materialize the fragment of an XML view needed by an application and it can fully exploit the query engine of a relational database management system whenever data items in an XML view need to be materialized.

49 Claims, 2 Drawing Sheets



The "helper " functions substX and substR perform the variable substitutions. Note that rewriteX calls compose recursively to rewrite a nested XML-QL block into an equivalent nested RXL block.

IV. Related Systems

IBM's DB2, XML Extender provides a Data Access Definition (DAD) language that supports both composition of relational data in XML and decomposition of XML data into relational tables. DAD's composition feature, like RXL, supports generation of arbitrary XML from relational data. Unlike RXL, the criteria for grouping elements is implicit in the DAD and DAD specifications cannot be nested arbitrarily. More significantly, XML Extender does not support query composition, however, DAD could be used as a view-definition language in a SilkRoute architecture.

V. General Discussion

SilkRoute is a general, dynamic, and efficient framework for viewing and querying relational data in XML. SilkRoute is an XML-export tool that can support arbitrarily complex, virtual views of relational data and support XML user queries over virtual views. The ability to support arbitrary views is critical in data exchange between inter-enterprise applications, which must abide by public DTDs and cannot reveal the underlying schemas of their proprietary data. SilkRoute has many benefits. For example, the fragment of the relational data requested by a user query need only be materialized; that requested data can be produced on demand; and the relational engine can perform most of the computation efficiently.

SilkRoute has one translation strategy, which generates one SQL query for each RXL sub query, which must be in disjunctive-normal form (DNF). In practice, RXL view queries can be arbitrary boolean combinations of table and filter expressions; for example, parallel. RXL blocks often construct parts of complex elements independently, i.e., they express unions. User queries over such views often produce composed queries with many unions. Any RXL sub-query can be normalized into multiple sub-queries in DNF, which can result in a quadratic increase in the number of sub-queries to evaluate. In practice, multiple queries in DNF can be translated directly into SQL, for example, by using SQL's union-join constructs. Similarly, nested RXL queries often express left outer joins, e.g., the parent sub-query is the left relation and the child sub-query is the right relation. Two SQL queries can be generated, one for parent and child, but one SQL query suffices. In addition to reducing the number of SQL queries, each individual RXL sub-query can be minimized, i.e., redundant expressions can be eliminated, so that the resulting SQL query is also, minimal. Techniques exist for query minimization, but general algorithms are NP-complete. Heuristic algorithms are projected to be effective for RXL queries, because RXL's nested block structure can help identify those expressions that most likely are redundant. XML-QL and SilkRoute can be implemented in Java. SilkRoute has drivers for Oracle and MySQL database servers.

What is claimed:

1. A method for converting relational data from a relational database into a structured document comprising the steps of:

storing a view query that defines a structured document view of the relational database, a structure of the view query being independent of a structure of data in the relational database;

receiving a user query against the structured document view;

forming an executable query by determining a composition of the view query and the user query;

partitioning the executable query into a data extraction portion and a construction portion;

transmitting the data extraction portion to the relational database;

receiving at least one tuple stream from the relational database according to the data extraction portion; and

merging the at least one tuple stream and the construction portion to generate a

structured document, wherein the structured document view is capable of defining a document of arbitrary nesting depth.

2. The method according to claim 1, wherein the user query is in XML-QL.

3. The method according to claim 2, wherein the view query is a RXL query.

4. The method according to claim 3, wherein the executable query is another RXL query.

5. The method device according to claim 1, wherein said step of receiving the user query includes receiving the user query from an application over a distributed network.

6. The method according to claim 5, wherein the distributed network is a private network.

7. The method according to claim 5, wherein the distributed network is the World Wide Web.

8. The method according to claim 5, further including the step of transmitting the structured document to the application over the distributed network.

9. The method according to claim 1, wherein an answer to the executable query includes a fragment of the relational database.

10. The method according to claim 9, wherein the fragment is one data item or a set of data items.

11. The method according to claim 9, wherein the fragment is an aggregate value.

12. The method according to claim 9, wherein the structured document includes the answer to the executable query.

13. The method according to claim 1, wherein the data extraction portion includes at least one SQL query.

14. The method according to claim 13, wherein the construction portion is an XML template.

15. The method according to claim 13, wherein said step of receiving the at least one tuple stream includes receiving one tuple stream for each SQL-query included in the data extraction portion.

16. The method according to claim 1, wherein said step of forming the executable query includes combining fragments of the user query and the view query.

17. A computer readable medium including a set of executable instructions stored thereon for converting relational data stored in a relational database into structured document data, the set of instructions including:

forming an executable query from a view query that defines a structured document view of the relational database and a user query, a structure of the view query being independent from a structure of the relational data in the relational database;

partitioning the executable query into a data extraction portion and a construction portion;

retrieving at least one tuple stream from the relational database according to the data extraction portion; and

merging the at least one tuple stream and the construction portion to generate a structured document, wherein

the structured document view is capable of defining a document of arbitrary nesting depth.

18. The computer readable medium according to claim 17, further comprising storage for the view query.

19. The computer readable medium according to claim 18, wherein the view query is a RXL query.

20. The computer readable medium according to claim 19, wherein the executable query is another RXL query.

21. The computer readable medium according to claim 17, wherein an answer to the executable query includes a fragment of the relational database.

22. The computer readable medium according to claim 21, wherein the fragment is one data item or a set of data items.

23. The computer readable medium according to claim 21, wherein the fragment is an aggregate value.

24. The computer readable medium according to claim 21, wherein the structured document includes the answer to the executable query.

25. The computer readable medium according to claim 17, wherein the data extraction portion includes at least one SQL query.

26. The computer readable medium according to claim 25, wherein said step of retrieving the at least one tuple stream includes retrieving one tuple stream for each SQL-query included in the data extraction portion.

27. The computer readable medium according to claim 17, wherein said instruction of forming the executable query includes combining fragments of the user query and the view query.

28. A computer system for converting relational data in a relational database into structured document data, said system comprising:

a query composer module configured to form an executable query from a view query that defines a structured document view of the relational data and a user query, the view query having a structure that is independent of a structure of the relational data in the relational database;

a translator module coupled to the query composer module configured to receive the executable query, partition the executable query into a data extraction portion and an construction portion, and transmit the data extraction portion to a relational database;

a generator module configured to receive the construction portion from said translator module, receive at least one tuple stream from the relational database based on data extraction portion, and merge the at least one tuple stream and the construction portion to generate a structured document, wherein

the structured document view is capable of defining a document of arbitrary nesting depth.

29. The computer system according to claim 28, wherein said translator module receives a description of a schema of the relational data prior to partitioning the executable query.

30. The computer system according to claim 28, wherein the user query is in XML-QL, the view query is a RXL query and the executable query is another RXL query.

31. The computer system according to claim 28, wherein said query composer module is configured to receive the user query from an application over a distributed network.

32. The computer system according to claim 31, wherein said generator module is configured to transmit the structured document to the application over the distributed network.

33. The computer system according to claim 28, wherein an answer to the executable query includes a fragment of the relational database.

34. The computer system according to claim 33, wherein the fragment is one data item, a set of data items or an aggregate value.

35. The computer system according to claim 33, wherein the structured document includes the answer to the executable query.

36. The computer system according to claim 28, wherein the data extraction portion includes at least one SQL query.

37. The computer system according to claim 36, wherein said generator module receives one tuple stream for each SQL-query included in the data extraction portion.

38. The computer system according to claim 28, wherein said query composer module combines fragments of the user query and the view query to form the executable query.

39. The computer system according to claim 28, wherein the data extraction portion is executable by the relational database.

40. The computer system according to claim 28, where in the query composer module includes

a pattern matcher module configured to match patterns of the user query and the view query to obtain a solutions relation, in which each tuple represents a match; and

a rewriter module configured to rewrite each tuple.

41. A method of forming a composed query for use in converting relational data of a relational database into XML data, said method comprising the steps of:

receiving a user query in an XML based language requesting relational data;

forming a view query in a transformation language, the view query defining a structured document view of the relational data and having a structure that is independent from a structure of the relational data of the relational database; and

composing an executable query in the transformation language from the view query and the user query, wherein

the structured document view is capable of defining a document of arbitrary nesting depth.

42. The method according to claim 41, wherein the transformation language is RXL.

43. The method according to claim 41, wherein said step of composing includes the steps of:

matching patterns of the user query and the view query to obtain a solutions relation, in which each tuple represents a match; and

rewriting each tuple into a clause in the transformation language.

44. The method according to claim 43, wherein said step, of matching patterns includes the steps of:

constructing a view tree from the view query; and

evaluating the user query on the view tree.

45. The method according to claim 43, wherein said step of rewriting includes the step of constructing at least one block using from and where clause of the view query and a construct clause of the user query.

46. The method according to claim 43, wherein said step of rewriting each tuple into a clause in the transformation language is conducted on the basis of from/where clauses of the view query, the solutions relation, and construct clauses of the user query.

47. The method according to claim 1, wherein a format of the structured document is XML.

48. The computer readable medium according to claim 17, wherein a format of the structured document is XML.

49. The computer system according to claim 28, wherein a format of the structured document generated by the generator module is XML.

* * * * *

65/7/17 (Item 17 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

0013059466 *Drawing available*
WPI Acc no: 2003-139207/200313
XRPX Acc No: N2003-110531
Relational database formation method involves assigning qualifiers to attribute and text, having possibility of repeating with another attribute and text, sharing common element and text, respectively
Patent Assignee: KRUPA K A (KRUP-I)
Inventor: KRUPA K A

PRO-SE ASSIGNEE

Patent Family (2 patents, 1 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
US 20020156811	A1	20021024	US 2000206325	P	20000523	200313	B
			US 2001863794	A	20010523		
US 6915304	B2	20050705	US 2001863794	A	20010523	200544	E

Priority Applications (no., kind, date): US 2000206325 P 20000523; US 2001863794 A 20010523

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
US 20020156811	A1	EN	14	5	Related to Provisional	US 2000206325

Alerting Abstract US A1
NOVELTY - A qualifier is assigned to an **attribute** that has a possibility of repeating with another **attribute** sharing a common element. Another qualifier is assigned to a text that has a possibility of repeating with another text, sharing a common **attribute**.
DESCRIPTION - An INDEPENDENT CLAIM is included for **relational database** structure.
USE - For forming **relational database** using an **XML data structure** in **software** technology.
ADVANTAGE - By assigning the qualifiers to the **attribute** and text, the requirement of recessive **querying** is avoided while maintaining the benefits of general **data** model, thus reducing the time of writing and testing **software**.
DESCRIPTION OF DRAWINGS - The figure shows a graphical depiction of an in-memory **XML document** structure.

Title Terms /Index Terms/Additional Words: RELATED; **DATABASE**; FORMATION; METHOD; ASSIGN; **ATTRIBUTE**; TEXT; POSSIBILITY; REPEAT; SHARE; COMMON; ELEMENT; RESPECTIVE

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date

G06F-015/00; G06F-017/00			Main		"Version 7"
G06F-017/30			Secondary		"Version 7"

US Classification, Issued: 707513000, 707514000, 707102000, 707104100, 707200000, 707203000

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): **T01-J05B4B**; **T01-J05B4M**; **T01-N03B2**

Original Publication Data by Authority

United States

Publication No. US 20020156811 A1 (Update 200313 B)
Publication Date: 20021024
System and method for converting an XML data structure into a relational database
Assignee: Krupa, Kenneth A., New York, NY, US (KRUP-I)
Inventor: Krupa, Kenneth A., New York, NY, US
Agent: COBRIN & GITTES, 750 Lexington Avenue, 21 Floor, New York, NY, US
Language: EN (14 pages, 5 drawings)
Application: US 2000206325 P 20000523 (Related to Provisional)
US 2001863794 A 20010523 (Local application)
Original IPC: G06F-15/00(A)
Current IPC: G06F-15/00(A)
Original US Class (main): 707513
Original US Class (secondary): 707514
Original Abstract: The present invention provides systems and methods for converting between an XML data structure and a relational database. It enables the storage of an XML document in such a way that: the relational data model would not have to change as the document model changes; the structure of the tables is set up in such a way that the entire document can be retrieved with a single query in a linear (i.e. non-recursive) fashion; and, information about specific individual components within an XML document can be retrieved via simple queries that do not require hierarchy traversals or intensive, post-query data parsing.
Claim: Having described the invention, what is claimed is:

1. 1. A method of forming a relational database, comprising mapping a corresponding unique key to each tree component of an Extensible Markup Language (XML) document, the mapping including forming each of the corresponding unique keys as associated tree strings, each of the associated tree strings including in corresponding hierarchical order derived from the tree components a parent, a child, and a descriptor, the parent being an element, the child being an attribute, and the descriptor being text;
 - assigning a qualifier to the child as warranted that has a possibility of repeating with another child sharing

the parent in common; and

- assigning a further qualifier to the descriptor as warranted that has a possibility of repeating with another descriptor sharing the child in common.

Publication No. US 6915304 B2 (Update 200544 E)
Publication Date: 20050705
System and method for converting an XML data structure into a relational database
Assignee: Krupa, Kenneth A., 278 E. 10th St., New York, NY 10009, US Residence: US (KRUP-I)
Inventor: Krupa, Kenneth A., 278 E. 10th St., New York, NY 10009, US Residence: US
Agent: Mintz Levin Cohn Ferris Glovsky & Popeo, P.C.
Language: EN
Application: US 2001863794 A 20010523 (Local application)
Original IPC: G06F-17/00(A) G06F-17/30(B)
Current IPC: G06F-17/00(A) G06F-17/30(B)
Original US Class (main): 707102
Original US Class (secondary): 707104.1 707200 707203
Original Abstract: The present invention provides systems and methods for converting between an XML data structure and a relational database. It enables the storage of an XML document in such a way that: the relational data model would not have to change as the document model changes; the structure of the tables is set up in such a way that the entire document can be retrieved with a single query in a linear (i.e. non-recursive) fashion; and, information about specific individual components within an XML document can be retrieved via simple queries that do not require hierarchy traversals or intensive, post-query data parsing.
Claim:

1. 1. A method of forming a relational database from an Extensible Markup Language (XML) document formed of a plurality of nodes, the method comprising:
 - assigning an identifier to every node of the XML document, wherein the identifier identifies the XML document against another XML document;
 - assigning a respective sequence identifier to each node of the XML document; wherein said respective sequence identifiers identify an order of the nodes in the XML document; and,
 - converting each node of the XML document into a respective row of the relational database.

US 20020156811A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2002/0156811 A1

Krupa (43) Pub. Date: Oct. 24, 2002

(54) SYSTEM AND METHOD FOR CONVERTING AN XML DATA STRUCTURE INTO A RELATIONAL DATABASE

(76) Inventor: Kenneth A. Krupa, New York, NY (US)

Correspondence Address:
COBRIN & GITTES
750 Lexington Avenue, 21 Floor
New York, NY 10022 (US)

(21) Appl. No.: 09/863,794

(22) Filed: May 23, 2001

Related U.S. Application Data

(60) Provisional application No. 60/206,325, filed on May 23, 2000.

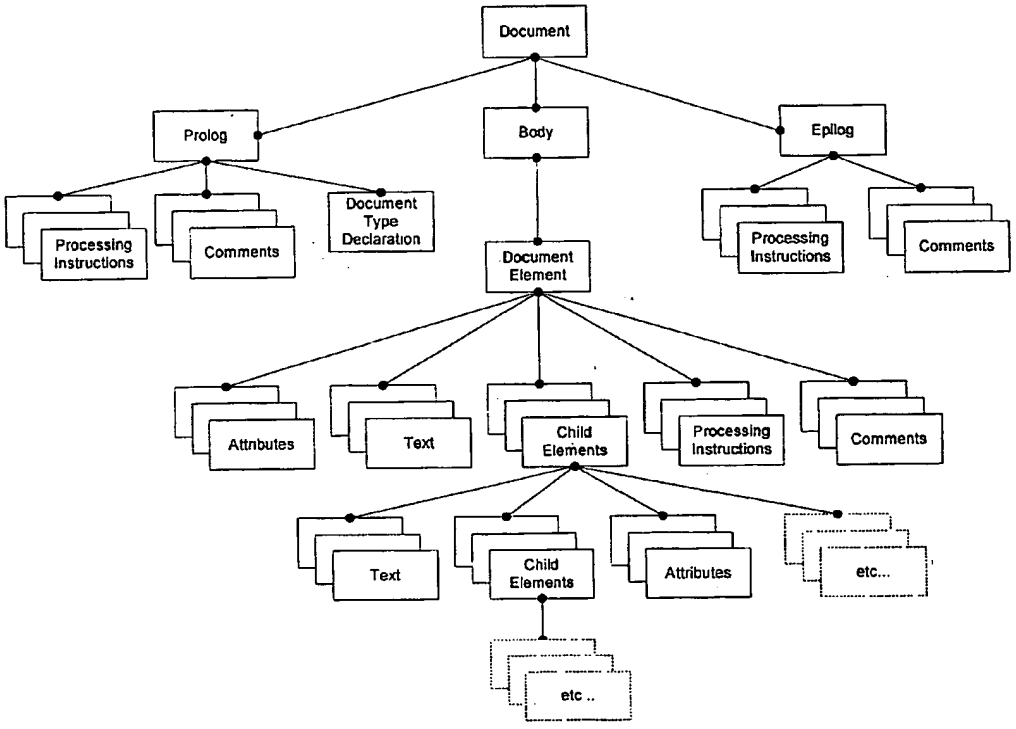
Publication Classification

(51) Int. Cl.[7] G06F 15/00

(52) U.S. Cl. 707/513; 707/514

(57) ABSTRACT

The present invention provides systems and methods for converting between an XML data structure and a relational database. It enables the storage of an XML document in such a way that: the relational data model would not have to change as the document model changes; the structure of the tables is set up in such a way that the entire document can be retrieved with a single query in a linear (i.e. non-recursive) fashion; and, information about specific individual components within an XML document can be retrieved via simple queries that do not require hierarchy traversals or intensive, post-query data parsing.



US006915304B2

(12) **United States Patent**
Krupa

(10) **Patent No.: US 6,915,304 B2**
(45) **Date of Patent: Jul. 5, 2005**

(54) **SYSTEM AND METHOD FOR CONVERTING AN XML DATA STRUCTURE INTO A RELATIONAL DATABASE**

(76) Inventor: **Kenneth A. Krupa**, 278 E. 10th St., New York, NY (US) 10009

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 293 days.

(21) Appl. No.: **09/863,794**

(22) Filed: **May 23, 2001**

(65) **Prior Publication Data**

US 2002/0156811 A1 Oct. 24, 2002

Related U.S. Application Data

(60) Provisional application No. 60/206,325, filed on May 23, 2000.

(51) **Int. Cl.**[7] **G06F 17/00**; G06F 17/30

(52) **U.S. Cl.** **707/102**; 707/104.1; 707/200; 707/203

(58) **Field of Search** 707/102, 104.1, 707/200, 203, 3, 5

(56) **References Cited**

U.S. PATENT DOCUMENTS

6,446,256 B1 * 9/2002 Hyman et al. 717/143
6,584,459 B1 * 6/2003 Chang et al. 707/3
6,711,590 B1 * 3/2004 Lennon 707/200
2001/0037345 A1 * 11/2001 Kiernan et al. 707/513
2001/0047372 A1 * 11/2001 Gorelik et al. 707/514
2001/0056429 A1 * 12/2001 Moore et al. 707/101
2002/0010764 A1 * 1/2002 Spicer 709/220
2002/0116371 A1 * 8/2002 Dodds et al. 707/3
2002/0123993 A1 * 9/2002 Chau et al. 707/5

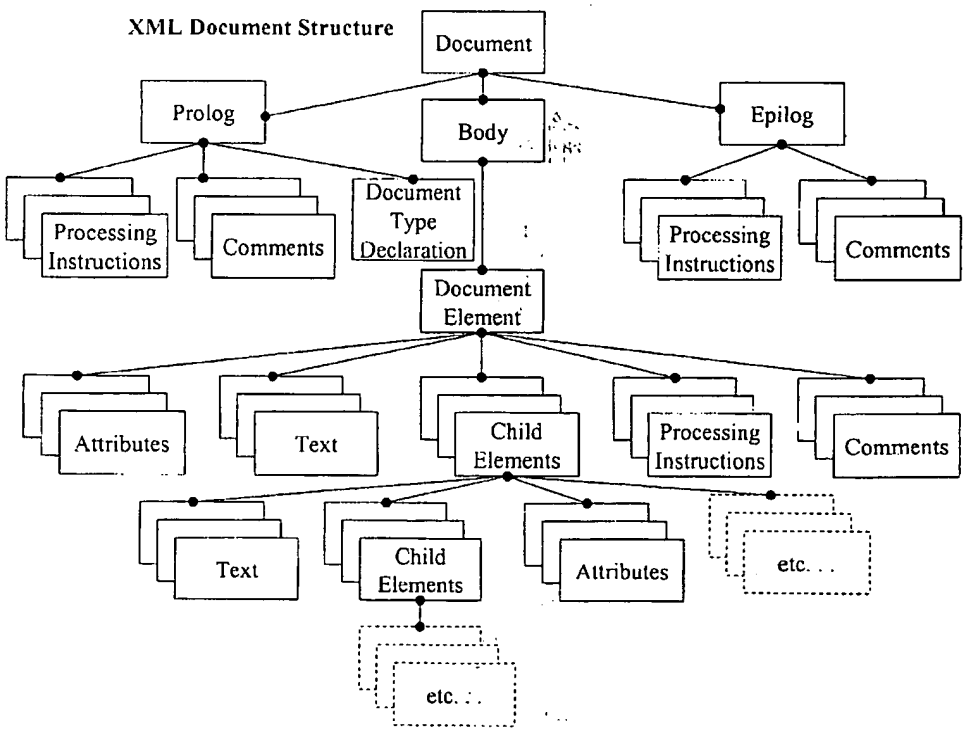
* cited by examiner

Primary Examiner—Luke Wassum
Assistant Examiner—Kuen S. Lu
(74) *Attorney, Agent, or Firm*—Mintz Levin Cohn Ferris Glovsky & Popeo, P.C.

(57) **ABSTRACT**

The present invention provides systems and methods for converting between an XML data structure and a relational database. It enables the storage of an XML document in such a way that: the relational data model would not have to change as the document model changes; the structure of the tables is set up in such a way that the entire document can be retrieved with a single query in a linear (i.e. non-recursive) fashion; and, information about specific individual components within an XML document can be retrieved via simple queries that do not require hierarchy traversals or intensive, post-query data parsing.

19 Claims, 6 Drawing Sheets



65/7/10 (Item 10 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

0014411948 *Drawing available*
WPI Acc no: 2004-601801/200458
XRPX Acc No: N2004-475796
Application-to-application communication managing system for use in distributed computer network, has commerce exchange server to determine message type and send extensible markup language documents for transaction
Patent Assignee: SUN MICROSYSTEMS INC (SUNM)
Inventor: ANKIREDDIPALLY L N; YEH R

Patent Family (1 patents, 1 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
US 6772216	B1	20040803	US 2000574334	A	20000519	200458	B

Priority Applications (no., kind, date): US 2000574334 A 20000519

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
US 6772216	B1	EN	29	20		

Alerting Abstract US B1
NOVELTY - The system has **extensible markup language (XML) application interaction documents** (40) for indicating a communication to one of requesting and service **applications**. The **XML document forms** an **application interaction** protocol. A commerce exchange server (10) determines a message type, and produces and sends the **XML documents** for transaction that is performed by a service **application** (30).
DESCRIPTION - The transaction response is **sent** to a **requesting application** (38). INDEPENDENT CLAIMS are also included for the following:

a. a method of managing **application**-to-**application** communication among set of **applications** and a set of service **applications** programs
b. an article of manufacture for a method of managing **application**-to-**application** communication in a distributed computer network
c. an **XML application interaction** message **document** stored on a computer-readable medium.

USE - Used for managing **application**-to-**application** communication among a set of **application** resident on a computer set in a distributed computer network (claimed) for a commercial transaction in a business.
ADVANTAGE - The **XML documents form** the underlying **application interaction** protocol, referred as a commerce exchange **interaction** protocol standardizing **application interaction**, thereby allowing straightforward integration of third party **applications** without the absolute requirement for **application**-specific libraries and hence

facilitating the automatic processing of today's complex commercial transactions. The **XML documents** provide a common set of generic **interactions** to enable full **application** interoperability in a distributed network environment.
DESCRIPTION OF DRAWINGS - The drawing shows a block diagram schematically illustrating system architecture for enabling **application**-to-**application interaction** in a distributed computer network.
10 Commerce exchange server
24 Computer
30 Service **application**
38 Requesting **application**
40 **XML application interaction documents**

Title Terms /Index Terms/Additional Words: APPLY; COMMUNICATE; MANAGE; SYSTEM; DISTRIBUTE; COMPUTER; NETWORK; EXCHANGE; SERVE; DETERMINE; MESSAGE; TYPE; SEND; EXTEND; LANGUAGE; DOCUMENT; TRANSACTION

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-015/173			Main		"Version 7"
G06F-015/16; G06F-009/00; G06F-009/46			Secondary		"Version 7"

US Classification, Issued: 709230000, 709227000, 709219000, 709203000, 719330000, 719328000

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): **T01-N02B1**; **T01-N03B2**; **T01-S03**

Original Publication Data by Authority

United States

Publication No. US 6772216 B1 (Update 200458 B)
Publication Date: 20040803
Interaction protocol for managing cross company processes among network-distributed applications
Assignee: Sun Microsystems, Inc., Palo Alto, CA, US (SUNM)
Inventor: Ankireddipally, Lakshmi Narasimha, Sunnyvale, CA, US
Yeh, Ryh-Wei, Palo Alto, CA, US
Agent: Hickman Palermo Truong & Becker LLP, US
Truong, Bobby K., US
Holmes, Craig G., US

Language: EN (29 pages, 20 drawings)
Application: US 2000574334 A 20000519 (Local application)
Original IPC: G06F-15/173(A) G06F-9/00(B) G06F-9/46(B) G06F-15/16(B)
Current IPC: G06F-15/173(A) G06F-9/00(B) G06F-9/46(B) G06F-15/16(B)
Original US Class (main): 709230
Original US Class (secondary): 709227 709219 709203 719330 719328
Original Abstract: A general purpose application interaction protocol, referred to herein as the Commerce Exchange Interaction Protocol (CXIP), governs the exchange of data between applications resident in computers in a distributed network such as the Internet, providing for application interoperability. CXIP enables process communications among network-distributed software applications, particularly, but not exclusively, among software applications having dissimilar platforms, language dependencies or vendor dependencies. Data structures, object and method invocation requests are exchanged between applications by means of XML documents based on CXIP semantics. The application interaction protocol, which is not specific to any particular functional domain and specifies interactions that are independent of transactional content, includes four component parts: message formats, message types, exchange semantics and transportation assumptions. CXIP-based interactions use TCP/IP as their underlying transport mechanism, although CXIP may be implemented using SMTP or FTP, or on top of any other standard application-layer protocol, including HTTP.
Claim: What is claimed is:

1. 1. A system for managing application-to-application communication among a plurality of applications resident on a plurality of computers in a distributed computer network, the system comprising:
 - at least one requesting application resident on one of the plurality of computers in the distributed computer network, the requesting application being capable of producing a transaction request indicating a transaction;
 - at least one service application resident on one of the plurality of computers in the distributed computer network, the service application being capable of performing the transaction and producing a transaction response;
 - a plurality of structured documents that each indicate a communication to one of the requesting application and the service application, wherein each structured document includes a message type that is associated with a particular predetermined sequence of structured documents of a plurality of predetermined sequences of structured documents, wherein the particular predetermined sequence of structured documents are to be exchanged between the requesting application and the service application, and wherein the plurality of predetermined sequences of structured documents collectively defines an application interaction protocol; and
 - a process automation application resident on one of the plurality of computers in the distributed computer network, the process automation application determining the message type and producing and sending the predetermined sequence of structured documents to cause the transaction to be performed by the service application and the transaction response to be sent to the requesting application.

US006772216B1

(12) **United States Patent**
Ankireddipally et al.

(10) **Patent No.: US 6,772,216 B1**
(45) **Date of Patent: *Aug. 3, 2004**

(54) **INTERACTION PROTOCOL FOR MANAGING CROSS COMPANY PROCESSES AMONG NETWORK-DISTRIBUTED APPLICATIONS**

(75) Inventors: **Lakshmi Narasimha Ankireddipally**, Sunnyvale, CA (US); **Ryh-Wei Yeh**, Palo Alto, CA (US)

(73) Assignee: **Sun Microsystems, Inc.**, Palo Alto, CA (US)

(*) Notice: This patent issued on a continued prosecution application filed under 37 CFR 1.53(d), and is subject to the twenty year patent term provisions of 35 U.S.C. 154(a)(2).

Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **09/574,334**

(22) Filed: **May 19, 2000**

(51) **Int. Cl.**[7] **G06F 15/173**; G06F 15/16; G06F 9/46; G06F 9/00

(52) **U.S. Cl.** **709/230**; 709/227; 709/219; 709/203; 719/330; 719/328

(58) **Field of Search** 709/230, 217, 709/219, 315, 330

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,893,108	A	4/1999	Srinivasan et al.	
5,950,212	A	9/1999	Anderson et al.	
5,956,483	A	9/1999	Grate et al.	395/200.33
5,970,472	A	10/1999	Allsop et al.	705/26
5,987,429	A	11/1999	Maritzen et al.	705/31
5,999,979	A	12/1999	Vellanki et al.	709/232
6,009,464	A	12/1999	Hamilton et al.	709/219
6,321,234	B1	11/2001	Debrunner	
6,457,066	B1 *	9/2002	Mein et al.	709/230
6,507,856	B1	1/2003	Chen et al.	
6,529,921	B1	3/2003	Berkowitz et al.	

OTHER PUBLICATIONS

Burns et al. ("Opentag and TMX: XML in the Localization Industry" Proceedings of the 16th annual international conference on Computer documentation; Quebec; pp. 137–142; Sep. 1998.*

"RosettaNet Implementation Framework Specification," Version 1.1 published Nov. 8, 1999, available at http://www.rosettanet.org as of Feb. 2000.

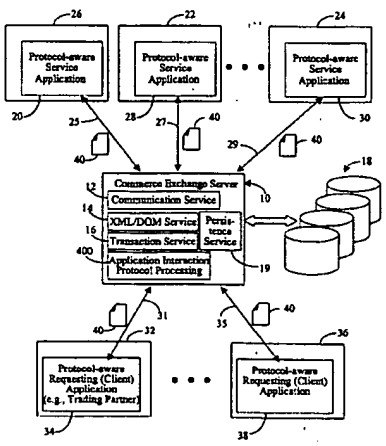
(List continued on next page.)

Primary Examiner—David Wiley
Assistant Examiner—Joseph E Avellino
(74) *Attorney, Agent, or Firm*—Hickman Palermo Truong & Becker LLP; Bobby K. Truong; Craig G. Holmes

(57) **ABSTRACT**

A general purpose application interaction protocol, referred to herein as the Commerce Exchange Interaction Protocol (CXIP), governs the exchange of data between applications resident in computers in a distributed network such as the Internet, providing for application interoperability. CXIP enables process communications among network-distributed software applications, particularly, but not exclusively, among software applications having dissimilar platforms, language dependencies or vendor dependencies. Data structures, object and method invocation requests are exchanged between applications by means of XML documents based on CXIP semantics. The application interaction protocol, which is not specific to any particular functional domain and specifies interactions that are independent of transactional content, includes four component parts: message formats, message types, exchange semantics and transportation assumptions. CXIP-based interactions use TCP/IP as their underlying transport mechanism, although CXIP may be implemented using SMTP or FTP, or on top of any other standard application-layer protocol, including HTTP.

24 Claims, 11 Drawing Sheets



TCP/IP protocols includes, but is not limited to, protocols such as TCP, IP, UDP, ICMP, FTP, and SMTP. Additional application and management protocols are sometimes considered part of the TCP/IP Suite. This includes protocols such as SNMP.

The World Wide Web ("Web") refers to a distributed collection of hypertext documents, also referred to as "Web documents" or "Web pages", that are accessible from a processor-controlled machine connected to the Internet using one or more standard Internet protocols. Web document may be viewed by a user or used by an application program. HTTP (defined below) is an example of a standard protocol for locating and acquiring Web documents. HTML (HyperText Markup Language) is an example of a language used to create Web documents.

The term client-server refers to a model of interaction in a distributed system in which a program at one site sends a request to a program at another site and waits for or obtains a response. The requesting program is called the "client," and the program that responds to the request is called the "server." One type of client is a "Web browser" (or simply "browser") which runs on a computer of a user; the program which responds to browser requests by serving Web pages is commonly referred to as a "Web server."

The Hypertext Transport Protocol, or HTTP, is, in the context of the World Wide Web, a client-server protocol that facilitates the transfer of hypertext-based files between local and remote systems. HTTP includes a number of different types of messages that may be sent from a client to a server to request different types of server actions. For example, HTTP governs the exchange of a client request for a HTML document from a Web browser and its receipt from a Web server.

MIME (Multipurpose Internet Multimedia Extensions) Type refers to a file extension or attachment that specifies the type or format of the file (e.g., HTML, text, graphics, audio, etc.). MIME typing allows a Web browser to determine how to process a file that is received from a Web server. Standard Web browsers come pre-configured with software interfaces known as plug-ins to handle popular MIME types, and can be configured by the user to handle new MIME types.

Extensible Markup Language, abbreviated XML, describes a class of data objects called XML documents and partially describes the behavior of computer programs which process them. XML is an application profile or restricted form of SGML, the Standard Generalized Markup Language [ISO 8879]. By construction, XML documents are conforming SGML documents. Each XML document has both a logical and a physical structure. Physically, the document is composed of units called entities. An entity may refer to other entities to cause their inclusion in the document. A document begins in a "root" or document entity. Logically, the document is composed of declarations, elements, comments, character references, and processing instructions, all of which are indicated in the document by explicit markup. The logical and physical structures must nest properly, as described in "4.3.2 Well-Formed Parsed Entities" in the World Wide Web Consortium XML specification. A software module called an XML processor is used to read XML documents and provide access to their content and structure. It is assumed that an XML processor is doing its work on behalf of another processing entity or module. An XML document is valid if it has an associated document type declaration (DTD) and if the document complies with the constraints expressed in its associated DTD. An XML document is a well-formed XML document if the document, taken as a whole, matches the XML production labeled "document," meets all the constraints with respect to being well-formed given in the XML specification, and each of the parsed entities referenced directly or indirectly within the document is well-formed. A well-formed XML document may also be valid if it meets additional criteria, as specified in World Wide Web Consortium, *Extensible Markup Language (XML) 1.0: W3C Recommendation* 10-Feb.-1998.) Additional information about XML is available at http://www.w3.org/XML and www.w3.org/TR/PR-xml-971208.

While the invention has been described in conjunction with one or more specific embodiments, this description is not intended to limit the invention in any way. Accordingly, the invention as described herein is intended to embrace all modifications and variations that are apparent to those skilled in the art and that fall within the scope of the appended claims.

What is claimed is:

1. A system for managing application-to-application communication among a plurality of applications resident on a plurality of computers in a distributed computer network, the system comprising:

at least one requesting application resident on one of the plurality of computers in the distributed computer network, the requesting application being capable of producing a transaction request indicating a transaction;

at least one service application resident on one of the plurality of computers in the distributed computer network, the service application being capable of performing the transaction and producing a transaction response;

a plurality of structured documents that each indicate a communication to one of the requesting application and the service application, wherein each structured document includes a message type that is associated with a particular predetermined sequence of structured documents of a plurality of predetermined sequences of structured documents, wherein the particular predetermined sequence of structured documents are to be exchanged between the requesting application and the service application, and wherein the plurality of predetermined sequences of structured documents collectively defines an application interaction protocol; and

a process automation application resident on one of the plurality of computers in the distributed computer network, the process automation application determining the message type and producing and sending the predetermined sequence of structured documents to cause the transaction to be performed by the service application and the transaction response to be sent to the requesting application.

2. The system of claim 1 for managing application-to-application communication among a plurality of applications resident on a plurality of computers in a distributed computer network wherein:

the message type is a particular message type of a plurality of message types;

the plurality of message types includes an acknowledgment message type defining an acknowledgement structured document indicating receipt of a first structured document previously sent by the process automation application;

the requesting application and the service application produce the acknowledgement structured document in response to receipt of a second structured document

that has any message type except an acknowledge message type; and

the process automation application monitors delivery of structured documents to requesting applications and service applications using the acknowledgement structured document.

3. The system of claim 1 for managing application-to-application communication among a plurality of applications resident on a plurality of computers in a distributed computer network wherein the process automation application provides for control of asynchronous interactions between the requesting application and the service application.

4. The system of claim 1 for managing application-to-application communication among a plurality of applications resident on a plurality of computers in a distributed computer network wherein the structured document uses Extensible Markup Language (XML) tags indicating control data and input-output data, the control data tag indicating the message type of the transaction message and a service type, the input-output data tag indicating input data required by the service type and output data produced by the service type.

5. The system of claim 1 for managing application-to-application communication among a plurality of applications resident on a plurality of computers in a distributed computer network wherein the process automation application sends the structured documents between the requesting application and the service application using the TCP/IP data communications protocol.

6. The system of claim 1 for application-to-application communication processing among a plurality of applications resident on a plurality of computers in a distributed computer network wherein the process automation application includes a protocol bridging operation for converting structured documents received from the requesting application using the TCP/IP data communications protocol to structured documents using the SMTP data communications protocol for transmission to the service application.

7. The system of claim 1 for application-to-application communication processing among a plurality of applications resident on a plurality of computers in a distributed computer network wherein the process automation application is a plug-in application associated with a Web browser in communication with the service application, and wherein the process automation application sends structured documents between the requesting application and the service application using the HTTP protocol.

8. The system of claim 1 for managing application-to-application communication among a plurality of applications resident on a plurality of computers in a distributed computer network wherein:

the message type is a particular message type of a plurality of message types; and

each of the plurality of message types indicates an expected action by an application that is independent of a functional domain of the transaction request.

9. The system of claim 1 for managing application-to-application communication among a plurality of applications resident on a plurality of computers in a distributed computer network wherein:

the message type is a particular message type of at least three message types;

each of the at least three message types is associated with one of a plurality of application-to-application interactions; and

the plurality of application-to-application interactions includes at least a request-reply application-to-application interaction, a publish-subscribe application-to-application interaction, and a broadcast-multicast application-to-application interaction.

10. In a distributed computer network, a method for managing application-to-application communication among a plurality of requesting applications and a plurality of service applications programs, the method comprising the steps of:

receiving transaction request data indicating a transaction from a requesting application;

determining a message type using the transaction request data, the message type being one of a plurality of message types, wherein the message type is associated with a particular predetermined sequence of structured documents of a plurality of predetermined sequences of structured documents, wherein the particular predetermined sequence of structured documents are to be exchanged between the requesting application and the service application, and wherein the plurality of predetermined sequences of structured documents collectively defines an application interaction protocol; and

producing, sending and receiving the particular predetermined sequence of structured documents, thereby causing the transaction to be performed by the service application and a transaction response to be sent to the requesting application.

11. The method of claim 10 for managing application-to-application communication among a plurality of requesting applications and a plurality of service applications wherein one of the structured documents in the particular predetermined sequence of structured documents is sent to the requesting application and has a particular message type of the plurality of message types indicating an acknowledgement of receipt of the transaction request data.

12. The method of claim 10 for managing application-to-application communication among a plurality of requesting applications and a plurality of service applications wherein the plurality of message types include Request, Reply, Publish, Notify, Subscribe and Unsubscribe.

13. The method of claim 10 for managing application-to-application communication among a plurality of requesting applications and a plurality of service applications wherein each structured document of the particular predetermined sequence of the structured documents uses Extensible Markup Language (XML) tags and has a document type definition (DTD) associated therewith.

14. The method of claim 10 for managing application-to-application communication among a plurality of requesting applications and a plurality of service applications wherein:

when the particular predetermined sequence of structured documents is produced according to a broadcast-multicast application-to-application interaction of the application interaction protocol, at least one of the structured documents is a Notify structured document having a message type of Notify and indicating transaction data to be sent to a subscriber list of service applications; and

the method further includes the step of selecting all service applications on the subscriber list from among the plurality of service applications, and sending the Notify structured document to the subscriber list of service applications to process the transaction data.

15. The method of claim 10 for managing application-to-application communication among a plurality of requesting applications and a plurality of service applications wherein:

when the particular predetermined sequence of structured documents is produced according to a publish-subscribe application-to-application interaction of the application interaction protocol, at least one of the structured documents is a Subscribe structured document having a message type of Subscribe and indicating transaction request data including a request to add the requesting application to a subscriber list of service applications; and

the method further includes the step of selecting at least one target service application from among the plurality of service applications, and sending the Subscribe structured document to the target service application to process the transaction request data.

16. The method of claim 10 for managing application-to-application communication among a plurality of requesting applications and a plurality of service applications wherein:

the message type is a particular message type of at least three message types;

each of the at least three message types is associated with one of a plurality of application-to-application interactions; and

the plurality of application-to-application interactions includes at least a request-reply application-to-application interaction, a publish-subscribe application-to-application interaction, and a broadcast-multicast application-to-application interaction.

17. A method for operating a first computer to manage application-to-application communication between a first application resident on the first computer and a second application resident on a second computer, the first and second computers being included in a distributed network, the method comprising the steps of:

receiving from the second application a request structured document having a request message type and indicating input application interaction data;

producing and sending to the second application a first acknowledge structured document, the first acknowledge structured document having an acknowledge message type and indicating an acknowledgement of receipt of the input application interaction data from the second application;

parsing the request structured document to produce first application-specific data;

invoking the first application to process the first application-specific data;

receiving output application-specific data indicating output results data from the first application processing the first application-specific data;

producing a reply structured document having a reply message type and including the output application-specific data;

sending the reply structured document to the second application resident on the second computer;

receiving from the second application a second acknowledge structured document having an acknowledge message type and indicating an acknowledgement of receipt of the output application interaction data from the first application;

wherein the steps above define a particular predetermined sequence of structured documents that are exchanged between the first application and the second application; and

wherein the particular predetermined sequence of structured documents is included in a plurality of predetermined sequences of structure documents that collectively defines an application interaction protocol.

18. The method of claim 17 for operating a first computer to manage application-to-application communication between a first application and a second application wherein each structured document of the request structured document, the reply structured document, the first acknowledge structured document, and the second acknowledge structured document uses Extensible Markup Language (XML) tags.

19. The method of claim 17 for operating a first computer to manage application-to-application communication between a first application and a second application wherein the first application is in communication with a Web browser application that has a plug-in application associated therewith for receiving and sending structured documents between the first application and the second application using the Hypertext Transport Protocol (HTTP) such that each structured document of the request structured document, the reply structured document, the first acknowledge structured document, and the second acknowledge structured document is transmitted using the HTTP protocol.

20. An article of manufacture comprising a data storage medium having computer readable instruction data embodied therein, the computer readable instruction data indicating instructions executed by a processor in a processor-controlled machine for managing application-to-application communication among a plurality of requesting applications and a plurality of service applications resident on a plurality of processor-controlled machines in a distributed network, the computer readable instructions in the article of manufacture comprising:

a first portion of instructions which when executed causes the processor to receive transaction request data indicating a transaction from a requesting application of the plurality of requesting applications;

a second portion of instructions which when executed causes the processor to determine a message type using the transaction request data, the message type being one of a plurality of message types, wherein the message type is associated with a particular predetermined sequence of structured documents of a plurality of predetermined sequences of structured documents, wherein the particular predetermined sequence of structured documents are to be exchanged between the requesting application and the service application, and wherein the plurality of predetermined sequences of structured documents collectively defines an application interaction protocol; and

a third portion of instructions which when executed causes the processor to produce, send and receive the particular predetermined sequence of structured documents, thereby causing the transaction to be performed by a service application of the plurality of service applications and a transaction response to be sent to the requesting application.

21. The article of manufacture of claim 20 wherein:

the message type is a particular message type of at least three message types;

each of the at least three message types is associated with one of a plurality of application-to-application interactions; and

the plurality of application-to-application interactions includes at least a request-reply application-to-

application interaction, a publish-subscribe application-to-application interaction, and a broadcast-multicast application-to-application interaction.

22. An XML application interaction message document stored on a computer-readable medium comprising:

a message header data portion indicating message-content size data; and

a message content data portion indicating a message type having one of a plurality of message types;

the message content data portion further indicating transaction request data;

the transaction request data, when parsed by a process automation application, causing the process automation application to produce a particular ordered sequence of XML application interaction message documents of a plurality of ordered sequences of structured documents that collectively defines an application interaction protocol, the process automation application controlling exchange of the particular ordered sequence of XML application interaction message documents between a requesting application and a service application.

23. The XML application interaction message document of claim 22 wherein the message content data portion further includes:

a service type specifying a service to be performed by the service application; and

an input-output data portion indicating input data required by the service application and output data produced by the service application.

24. The XML application interaction message document stored on a computer-readable medium of claim 22 wherein:

the plurality of message types includes at least three message types;

each of the at least three message types is associated with one of a plurality of application-to-application interactions; and

the plurality of application-to-application interactions includes at least a request-reply application-to-application interaction, a publish-subscribe application-to-application interaction, and a broadcast-multicast application-to-application interaction.

* * * * *

65/7/43 (Item 43 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

0010742716 *Drawing available*
WPI Acc no: 2001-355381/200137
XRPX Acc No: N2001-258237
Shared Internet storage system has client application which controls stored files and file status information, based on meta database information
Patent Assignee: HALD M (HALD-I); MCGREGOR L (MCGR-I); OBRIEN B (OBRI-I); WHITELEY S (WHIT-I); XDRIVE INC (XDRI-N); XDRIVE LLC (XDRI-N); XDRIVE TECHNOLOGIES INC (XDRI-N)
Inventor: HALD M; MCGREGOR L; O'BRIEN B; OBRIEN B; WHITELEY S

Patent Family (10 patents, 92 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
WO 2001033381	A1	20010510	WO 2000US30536	A	20001103	200137	B
AU 200114683	A	20010514	AU 200114683	A	20001103	200149	E
US 6351776	B1	20020226	US 1999163626	P	19991104	200220	E
			US 2000570583	A	20000512		
US 20020133561	A1	20020919	US 1999163626	P	19991104	200264	E
			US 2000570583	A	20000512		
			US 200273775	A	20020211		
JP 2003514279	W	20030415	WO 2000US30536	A	20001103	200328	E
			JP 2001535806	A	20001103		
US 20050171956	A1	20050804	US 1999163626	P	19991104	200552	E
			US 2000570583	A	20000512		
			US 200273775	A	20020211		
			US 200589397	A	20050324		
US 20050193132	A1	20050901	US 1999163626	P	19991104	200558	E
			US 2000570583	A	20000512		
			US 200273775	A	20020211		
			US 200575311	A	20050308		
US 20050204020	A1	20050915	US 1999163626	P	19991104	200561	E
			US 2000570583	A	20000512		
			US 200273775	A	20020211		
			US 200589174	A	20050324		
US 20050204063	A1	20050915	US 1999163626	P	19991104	200561	E
			US 2000570583	A	20000512		
			US 200273775	A	20020211		
			US 200589173	A	20050323		
US 6985927	B2	20060110	US 1999163626	P	19991104	200607	E
			US 2000570583	A	20000512		
			US 200273775	A	20020211		

					Division of application	US 200273775
					Continuation of patent	US 6351776
US 6985927	B2	EN			Related to Provisional	US 1999163626
					Continuation of application	US 2000570583
					Continuation of patent	US 6351776

Alerting Abstract WO A1

NOVELTY - A pair of databases stores file data and meta data which contains information about stored file data. A server communicating with a client application simultaneously executes file commands of file database and updates meta database, based on the file commands invoked by client application. The client application controls files and file status information, based on meta database.
DESCRIPTION - INDEPENDENT CLAIMS are also included for the following:

A. Public computer network private file space and information transfer providing method;
B. File operations effecting data structure;
C. Computer network shared file storage resource;
D. Network resources data transferring method;
E. Client-server system

USE - For allowing direct Internet-to-Internet file transfer to allocated user's file space.
ADVANTAGE - Allows users to store and retrieve files by themselves, throughout the Internet by controlling the stored files and file status information using client application.
DESCRIPTION OF DRAWINGS - The figure shows the schematic view of X drive unit of shared computer network storage system.

Title Terms /Index Terms/Additional Words: SHARE; STORAGE; SYSTEM; CLIENT; APPLY; CONTROL; FILE; STATUS; INFORMATION; BASED; META; DATABASE

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-012/00; G06F-015/00; G06F-015/16; G06F-015/167; G06F-007/00			Main		"Version 7"
B41B-015/00; G06F-011/00; G06F-013/00; G06F-017/30			Secondary		"Version 7"
G06F-0015/167	A	I	F	B	20060101
G06F-0015/16	C	I	F	B	20060101

Inventor: O'Brien, Brett, Pacific Palisades, CA, US
Whiteley, Sean, Millbrae, CA, US
McGregor, Lucas, Santa Monica, CA, US
Hald, Martin, Irvine, CA, US
Agent: CISLO & THOMAS, LLP, 233 WILSHIRE BLVD, SUITE 900, SANTA MONICA, CA, US
Language: EN
Application: US 1999163626 P 19991104 (Related to Provisional)
US 2000570583 A 20000512 (Continuation of application)
US 200273775 A 20020211 (Local application)
Related Publication: US 6351776 A (Continuation of patent)
Original IPC: G06F-15/167(A)
Current IPC: G06F-15/167(A)
Original US Class (main): 709213
Original US Class (secondary): 709218
Original Abstract: The Shared Internet Storage Resource provides Internet-based file storage, retrieval, access, control, and manipulation for a user. Additionally, an easy-to-use user interface is provided both for a browser or stand-alone application. The entire method provides means by which users can establish, use, and maintain files on the Internet in a manner remote from their local computers yet in a manner that is similar to the file manipulation used on their local computers. A high capacity or other storage system is attached to the Internet via an optional internal network that also serves to generate and direct metadata regarding the stored files. A web server using a CGI, Java(R)-based, or other interface transmits and retrieves TCP/IP packets or other Internet information through a load balancer/firewall by using XML to wrap the data packets. File instructions may be transmitted over the Internet to the Shared Resource System. The user's account may be password protected so that only the user may access his or her files. On the user's side, a stand-alone client application or JavaScript object interpreted through a browser provide two means by which the XML or other markup language data stream may be received and put to use by the user. Internet-to-Internet file transfers may be effected by directly downloading to the user's account space.
Claim: What is claimed is:

1. 1. A shared computer network storage system, comprising:
- a first database containing file data;
- a second database containing information (metadata) about said file data of said first database;
- a server, said server executing file commands on said first file database, said server contemporaneously updating said second metadatabase upon executing said file commands; and
- a client application, said client application communicating with said server, said client application invoking file commands upon said server, said server executing said file commands and updating information regarding said first file and second metadata databases displayed by said client application; whereby
- said client application controls files in said first file database and information regarding status of said first database files is more readily available by reference to said second metadatabase.

Publication No. US 20050171956 A1 (Update 200552 E)
Publication Date: 20050804
Shared internet storage resource, user interface system, and method
Assignee: OBRIEN B; US (OBRI-I)
Whiteley, Sean, Millbrae, CA, US Residence: US Nationality: US (WHIT-I)
Mcgregor, Lucas, Santa Monica, CA, US Residence: US Nationality: US (MCGR-I)

Hald, Martin, Irvine, CA, US Residence: US Nationality: US (HALD-I)
O'Brien, Brett, Pacific Palisades, CA, US Residence: US Nationality: US
Inventor: OBRIEN B
Whiteley, Sean, Millbrae, CA, US Residence: US Nationality: US
Mcgregor, Lucas, Santa Monica, CA, US Residence: US Nationality: US
Hald, Martin, Irvine, CA, US Residence: US Nationality: US
Agent: CISLO & THOMAS, LLP, 233 WILSHIRE BLVD, SUITE 900, SANTA MONICA, CA, US
Language: EN
Application: US 1999163626 P 19991104 (Related to Provisional)
US 2000570583 A 20000512 (Continuation of application)
US 200273775 A 20020211 (C-I-P of application)
US 200589397 A 20050324 (Local application)
Related Publication: US 6351776 A (Continuation of patent)
Original IPC: G06F-7/00(A)
Current IPC: G06F-7/00(A)
Original US Class (main): 70710
Original Abstract: The Shared Internet Storage Resource provides Internet-based file storage, retrieval, access, control, and manipulation for a user. Additionally, an easy-to-use user interface is provided both for a browser or stand-alone application. The entire method provides means by which users can establish, use, and maintain files on the Internet in a manner remote from their local computers yet in a manner that is similar to the file manipulation used on their local computers. A high capacity or other storage system is attached to the Internet via an optional internal network that also serves to generate and direct metadata regarding the stored files. A web server using a CGI, Java(R)-based, or other interface transmits and retrieves TCP/IP packets or other Internet information through a load balancer/firewall by using XML to wrap the data packets. File instructions may be transmitted over the Internet to the Shared Resource System. The user's account may be password protected so that only the user may access his or her files. On the user's side, a stand-alone client application or JavaScript object interpreted through a browser provide two means by which the XML or other markup language data stream may be received and put to use by the user. Internet-to-Internet file transfers may be effected by directly downloading to the user's account space.
Claim:

1. 1. A client-server system for a network-based data storage and manipulation system, comprising:
 - a client system, said client system having a file access service and a file manipulation service;
 - a server, said server providing network-based data storage resources and responding to requests transmitted by said client system, said server effecting said requests;
 - said server determining if a client request is one for metadata regarding data stored upon said server;
 - said server providing said metadata if said client request is for metadata and transmitting said metadata to said file manipulation service; and
 - said server performing a file action if said client request is not for metadata, said server updating said metadata and transmitting said metadata to said file manipulation service; whereby
 - said server operates, and said client system presents, operations on said server in a manner similar to operations local to said client system.

Publication No. US 20050193132 A1 (Update 200558 E)
Publication Date: 20050901
Shared internet storage resource, user interface system, and method

Assignee: OBRIEN B; US (OBRI-I)
Whiteley, Sean, Millbrae, CA, US Residence: US Nationality: US (WHIT-I)
McGregor, Lucus, Santa Monica, CA, US Residence: US Nationality: US (MCGR-I)
Hald, Martin, Irvine, CA, US Residence: US Nationality: US (HALD-I)
O'Brien, Brett, Pacific Palisades, CA, US Residence: US Nationality: US
Inventor: OBRIEN B
Whiteley, Sean, Millbrae, CA, US Residence: US Nationality: US
McGregor, Lucus, Santa Monica, CA, US Residence: US Nationality: US
Hald, Martin, Irvine, CA, US Residence: US Nationality: US
Agent: CISLO & THOMAS, LLP, 233 WILSHIRE BLVD, SUITE 900, SANTA MONICA, CA, US
Language: EN
Application: US 1999163626 P 19991104 (Related to Provisional)
US 2000570583 A 20000512 (Continuation of application)
US 200273775 A 20020211 (Division of application)
US 200575311 A 20050308 (Local application)
Related Publication: US 6351776 A (Continuation of patent)
Original IPC: G06F-15/16(A) G06F-11/00(B)
Current IPC: G06F-15/16(A) G06F-11/00(B)
Original US Class (main): 709229
Original US Class (secondary): 7145
Original Abstract: The Shared Internet Storage Resource provides Internet-based file storage, retrieval, access, control, and manipulation for a user. Additionally, an easy-to-use user interface is provided both for a browser or stand-alone application. The entire method provides means by which users can establish, use, and maintain files on the Internet in a manner remote from their local computers yet in a manner that is similar to the file manipulation used on their local computers. A high capacity or other storage system is attached to the Internet via an optional internal network that also serves to generate and direct metadata regarding the stored files. A web server using a CGI, Java(R)-based, or other interface transmits and retrieves TCP/IP packets or other Internet information through a load balancer/firewall by using XML to wrap the data packets. File instructions may be transmitted over the Internet to the Shared Resource System. The user's account may be password protected so that only the user may access his or her files. On the user's side, a stand-alone client application or JavaScript object interpreted through a browser provide two means by which the XML or other markup language data stream may be received and put to use by the user. Internet-to-Internet file transfers may be effected by directly downloading to the user's account space.

Claim:

1. **1-22.** (canceled)

Publication No. US 20050204020 A1 (Update 200561 E)
Publication Date: 20050915
Shared internet storage resource, user interface system, and method
Assignee: OBRIEN B; US (OBRI-I)
Whiteley, Sean, Millbrae, CA, US Residence: US Nationality: US (WHIT-I)
McGregor, Lucas, Santa Monica, CA, US Residence: US Nationality: US (MCGR-I)
Hald, Martin, Irvine, CA, US Residence: US Nationality: US (HALD-I)
O'Brien, Brett, Pacific Palisades, CA, US Residence: US Nationality: US

Inventor: OBRIEN B
Whiteley, Sean, Millbrae, CA, US Residence: US Nationality: US
McGregor, Lucas, Santa Monica, CA, US Residence: US Nationality: US
Hald, Martin, Irvine, CA, US Residence: US Nationality: US
Agent: CISLO & THOMAS, LLP, 233 WILSHIRE BLVD, SUITE 900, SANTA MONICA, CA, US
Language: EN
Application: US 1999163626 P 19991104 (Related to Provisional)
US 2000570583 A 20000512 (Continuation of application)
US 200273775 A 20020211 (Continuation of application)
US 200589174 A 20050324 (Local application)
Related Publication: US 6351776 A (Continuation of patent)
Original IPC: G06F-15/16(A) G06F-17/30(B)
Current IPC: G06F-15/16(A) G06F-17/30(B)
Original US Class (main): 709219
Original Abstract: The Shared Internet Storage Resource provides Internet-based file storage, retrieval, access, control, and manipulation for a user. Additionally, an easy-to-use user interface is provided both for a browser or stand-alone application. The entire method provides means by which users can establish, use, and maintain files on the Internet in a manner remote from their local computers yet in a manner that is similar to the file manipulation used on their local computers. A high capacity or other storage system is attached to the Internet via an optional internal network that also serves to generate and direct metadata regarding the stored files. A web server using a CGI, Java(R)-based, or other interface transmits and retrieves TCP/IP packets or other Internet information through a load balancer/firewall by using XML to wrap the data packets. File instructions may be transmitted over the Internet to the Shared Resource System. The user's account may be password protected so that only the user may access his or her files. On the user's side, a stand-alone client application or JavaScript object interpreted through a browser provide two means by which the XML or other markup language data stream may be received and put to use by the user. Internet-to-Internet file transfers may be effected by directly downloading to the user's account space.
Claim:

1. 1. A method for transferring data from a first network resource to a second network resource at the direction of a user, the steps comprising:
 - submitting a first file location indicating data to be transferred to the second network resource;
 - the second network resource requesting said data at said first file location from the first network resource;
 - the first network resource transmitting said data to the second network resource; and
 - the second network resource notifying the user of successful transfer upon successful reception of said data; whereby
 - the user may use the first and second network resources to obtain and control said data.

Publication No. US 20050204063 A1 (Update 200561 E)
Publication Date: 20050915
Shared internet storage resource, user interface system, and method
Assignee: OBRIEN B; US (OBRI-I)
Whiteley, Sean, Millbrae, CA, US Residence: US Nationality: US (WHIT-I)
McGregor, Lucas, Santa Monica, CA, US Residence: US Nationality: US (MCGR-I)
Hald, Martin, Irvine, CA, US Residence: US Nationality: US (HALD-I)
O'Brien, Brett, Pacific Palisades, CA, US Residence: US Nationality: US

Inventor: OBRIEN B
Whiteley, Sean, Millbrae, CA, US Residence: US Nationality: US
McGregor, Lucas, Santa Monica, CA, US Residence: US Nationality: US
Hald, Martin, Irvine, CA, US Residence: US Nationality: US
Agent: CISLO & THOMAS, LLP, 233 WILSHIRE BLVD, SUITE 900, SANTA MONICA, CA, US
Language: EN
Application: US 1999163626 P 19991104 (Related to Provisional)
US 2000570583 A 20000512 (Continuation of application)
US 200273775 A 20020211 (Division of application)
US 200589173 A 20050323 (Local application)
Related Publication: US 6351776 A (Continuation of patent)
Original IPC: G06F-15/16(A)
Current IPC: G06F-15/16(A)
Original US Class (main): 709245
Original Abstract: The Shared Internet Storage Resource provides Internet-based file storage, retrieval, access, control, and manipulation for a user. Additionally, an easy-to-use user interface is provided both for a browser or stand-alone application. The entire method provides means by which users can establish, use, and maintain files on the Internet in a manner remote from their local computers yet in a manner that is similar to the file manipulation used on their local computers. A high capacity or other storage system is attached to the Internet via an optional internal network that also serves to generate and direct metadata regarding the stored files. A web server using a CGI, Java(R)-based, or other interface transmits and retrieves TCP/IP packets or other Internet information through a load balancer/firewall by using XML to wrap the data packets. File instructions may be transmitted over the Internet to the Shared Resource System. The user's account may be password protected so that only the user may access his or her files. On the user's side, a stand-alone client application or JavaScript object interpreted through a browser provide two means by which the XML or other markup language data stream may be received and put to use by the user. Internet-to-Internet file transfers may be effected by directly downloading to the user's account space.
Claim:

1. 1-33. (canceled)

Publication No. US 6351776 B1 (Update 200220 E)
Publication Date: 20020226
Shared internet storage resource, user interface system, and method.
Assignee: XDrive, Inc., Santa Monica, CA, US
Inventor: OprimeBrien, Brett, Pacific Palisades, CA, US
Whiteley, Sean, Millbrae, CA, US
McGregor, Lucas, Santa Monica, CA, US
Hald, Martin, Irvine, CA, US
Agent: Cislo & Thomas, LLP
Language: EN
Application: US 1999163626 P 19991104 (Related to Provisional)
US 2000570583 A 20000512 (Local application)
Original IPC: G06F-15/16(A)
Current IPC: G06F-15/16(A)
Original US Class (main): 709245

Original US Class (secondary): 709223 709227 709247 7071 7074 70710 707100 707104
Original Abstract: The Shared Internet Storage Resource provides Internet-based file storage, retrieval, access, control, and manipulation for a user. Additionally, an easy-to-use user interface is provided both for a browser or stand-alone application. The entire method provides means by which users can establish, use, and maintain files on the Internet in a manner remote from their local computers yet in a manner that is similar to the file manipulation used on their local computers. A high capacity or other storage system is attached to the Internet via an optional internal network that also serves to generate and direct metadata regarding the stored files. A web server using a CGI, Java(R)-based, or other interface transmits and retrieves TCP/IP packets or other Internet information through a load balancer/firewall by using XML to wrap the data packets. File instructions may be transmitted over the Internet to the Shared Resource System. The userprimes account may be password protected so that only the user may access his or her files. On the userprimes side, a stand-alone client application or JavaScript object interpreted through a browser provide two means by which the XML or other markup language data stream may be received and put to use by the user. Internet-to-Internet file transfers may be effected by directly downloading to the userprimes account space.
Claim:

1. A client-server system for a network-based data storage and manipulation system, comprising:
 - a client system, said client system having a file access service and a file manipulation service;
 - a server, said server providing network-based data storage resources, said server creating and maintaining metadata regarding stored data, said server responding to requests transmitted by said client system, said server effecting said requests;
 - said server determining if a client request is one for metadata;
 - said server providing said metadata if said client request is for metadata and transmitting said metadata to said file manipulation service;
 - said server performing a file action if said client request is not for metadata, said server updating said metadata and transmitting said metadata to said file manipulation service;
 - said file access service having a request processing layer for processing requests and a first network I/O layer for transmitting said requests to said server;
 - said file manipulation service having an XML parser, said XML parser parsing said metadata from said server, said file manipulation service having a data structure, said data structure receiving and preserving parsed data from said parser, and said file manipulation service having a data display layer, said data display layer operating upon and displaying said parsed data so that metadata may be displayed to inform about data stored upon said server; and
 - said server having a second network I/O layer, said second network I/O layer engaged when said requests are not for metadata, said second network I/O layer transmitting requests for file action, said server having a resource access layer, said resource access layer receiving transmissions from said second network I/O layer and effecting said requests, said resource access layer engaged when said requests are for metadata, said resource access layer obtaining and transmitting said metadata, and said server having a metadata compiler in the form of an XML generator, said metadata compiler receiving said metadata from said resource access layer, compiling said metadata, and transmitting said compiled metadata to said client system; whereby
 - said server operates as apparently local to said client system and said client system presents operations on said server in a manner similar to operations local to said client system.

Publication No. US 6985927 B2 (Update 200607 E)
Publication Date: 20060110
Shared internet storage resource, user interface system, and method

Assignee: XDrive, LLC., Dulles, VA, US (XDRI-N)
O'Brien, Brett, Pacific Palisades, CA, US Residence: US
Whiteley, Sean, Millbrae, CA, US Residence: US
McGregor, Lucas, Santa Monica, CA, US Residence: US
Hald, Martin, Irvine, CA, US Residence: US
Inventor: OBRIEN B
Whiteley, Sean, Millbrae, CA, US Residence: US
McGregor, Lucas, Santa Monica, CA, US Residence: US
Hald, Martin, Irvine, CA, US Residence: US
Agent: Cislo & Thomas, LLP
Language: EN
Application: US 1999163626 P 19991104 (Related to Provisional)
US 2000570583 A 20000512 (Continuation of application)
US 200273775 A 20020211 (Local application)
Related Publication: US 6351776 A (Continuation of patent)
Original IPC: G06F-15/16(B,I,H,98,20060101,20060110,C,F) G06F-15/167(B,I,H,US,20060101,20060110,A,F)
Current IPC: G06F-15/16(B,I,H,98,20060101,20060110,C,F) G06F-15/167(B,I,H,US,20060101,20060110,A,F)
Original US Class (main): 709213
Original US Class (secondary): 709218 7071 7078 70710 707100 707201
Original Abstract: The Shared Internet Storage Resource provides Internet-based file storage, retrieval, access, control, and manipulation for a user. Additionally, an easy-to-use user interface is provided both for a browser or stand-alone application. The entire method provides means by which users can establish, use, and maintain files on the Internet in a manner remote from their local computers yet in a manner that is similar to the file manipulation used on their local computers. A high capacity or other storage system is attached to the Internet via an optional internal network that also serves to generate and direct metadata regarding the stored files. A web server using a CGI, Java(R)-based, or other interface transmits and retrieves TCP/IP packets or other Internet information through a load balancer/firewall by using XML to wrap the data packets. File instructions may be transmitted over the Internet to the Shared Resource System. The user's account may be password protected so that only the user may access his or her files. On the user's side, a stand-alone client application or JavaScript object interpreted through a browser provide two means by which the XML or other markup language data stream may be received and put to use by the user. Internet-to-Internet file transfers may be effected by directly downloading to the user's account space.
Claim: What is claimed is:

1. 1. A shared computer network storage system, comprising:
 - a first database containing file data;
 - a second database containing information (metadata) about said file data of said first database;
 - a server, said server executing file commands on said first file database, said server contemporaneously updating said second metadatabase upon executing said file commands, said server comprising:
 - a transaction processor, said transaction processor on a non-routable network, said transaction processor guarantees access to and transactions on said first and second databases;
 - an enterprise java bean cluster (EJBC) on a non-routable network, said enterprise java bean cluster (EJBC) coupled to said transaction processor and handling business logic and resource access methods as well as memory caching for common resources;
 - an application network on a non-routable network, said application network coupled to said enterprise java bean cluster, said application network including a java application cluster and handling display functions and resource requests;

- a web server, said web server coupled to said application network and handling all requests for static content and proxies requests for dynamic content;
- a load balancer, said load balancer coupled to said web server and proxying requests to a sub-server having the highest degree of availability or functionality; and
- a DNS redirector, said DNS redirector coupled to said load balancer and proxying requests to a resource having a highest degree of functionality; and

- a client application, said client application communicating with said server, said client application invoking file commands upon said server, said server executing said file commands and updating information regarding said first file and second metadata databases displayed by said client application; whereby
- said client application controls files in said first file database and information regarding status of said first database files is more readily available by reference to said second metadatabase.

WIPO

Publication No. WO 2001033381 A1 (Update 200137 B)
Publication Date: 20010510
SHARED INTERNET STORAGE RESOURCE, USER INTERFACE SYSTEM, AND METHOD
RESSOURCE DE STOCKAGE INTERNET PARTAGEE, SYSTEME D'INTERFACE UTILISATEUR ET PROCEDE CORRESPONDANT
Assignee: *(except US)* XDRIVE, INC, Attn: Legal Dept., 3002 Pennsylvania Avenue, Santa Monica, CA 90404, US Residence: US Nationality: US (XDRI-N)
(only US) O'BRIEN, Brett, XDrive, Inc., 3002 Pennsylvania Avenue, Santa Monica, CA 90404, US Residence: US Nationality: US
(only US) WHITELEY, Sean, XDrive, Inc., 3002 Pennsylvania Avenue, Santa Monica, CA 90404, US Residence: US Nationality: US
(only US) MCGREGOR, Lucas, XDrive, Inc., 3002 Pennsylvania Avenue, Santa Monica, CA 90404, US Residence: US Nationality: US
(only US) HALD, Martin, XDrive, Inc., 3002 Pennsylvania Avenue, Santa Monica, CA 90404, US Residence: US Nationality: DK
Inventor: O'BRIEN, Brett, XDrive, Inc., 3002 Pennsylvania Avenue, Santa Monica, CA 90404, US Residence: US Nationality: US
WHITELEY, Sean, XDrive, Inc., 3002 Pennsylvania Avenue, Santa Monica, CA 90404, US Residence: US Nationality: US
MCGREGOR, Lucas, XDrive, Inc., 3002 Pennsylvania Avenue, Santa Monica, CA 90404, US Residence: US Nationality: US
HALD, Martin, XDrive, Inc., 3002 Pennsylvania Avenue, Santa Monica, CA 90404, US Residence: US Nationality: DK
Agent: JORDAN, Andrew, Cislo & Thomas LLP, Suite 900, 233 Wilshire Boulevard, Santa Monica, CA 90401-1211, US
Language: EN (307 pages, 14 drawings)
Application: WO 2000US30536 A 20001103 (Local application)
Priority: US 1999163626 P 19991104
US 2000570583 A 20000512
Designated States: (National Original) AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CR CU CZ DE DK DM DZ EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV

MA MD MG MK MN MW MX MZ NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM TR TT TZ UA UG US UZ VN YU ZA ZW
(Regional Original) AT BE CH CY DE DK EA ES FI FR GB GH GM GR IE IT KE LS LU MC MW MZ NL OA PT SD SE SL SZ TR TZ UG ZW
Original IPC: G06F-15/00(A) B41B-15/00(B) G06F-15/16(B) G06F-17/30(B)
Current IPC: G06F-15/00(A) B41B-15/00(B) G06F-15/16(B) G06F-17/30(B)
Original Abstract: The shared internet storage resources provides Internet-based file storage, retrieval, access, control, and manipulation for user. Additionally, an easy-to-use user interface is provided both for a browser or stand-alone application. The entire method provides means by which users can establish, use, and maintain files on the internet in a manner remote from their local computers yet in a manner that is similar to the file manipulation used on their local computers. A high capacity or other storage system is attached to the internet via an optional internet network that also serves to generate and direct metadata regarding the stored files. A web server (140) using a CGI, Java-based, or other interface transmits and retrieves TCP/IP packets or other internet information through a load balancer/firewall (110) by using XML to wrap the data packets. File instructions may be transmitted over the Internet to the Shared Resource System. The user's account may be password protected so that only the user may access his or her files. On the user's side, a stand-alone client application (142) or JavaScript object interpreted through a browser provide two means by which the XML,or other markup language data stream may be received and put to use by the user. Internet-to-internet file transfer may be effected by directly downloading to the user's account space.

65/7/46 (Item 46 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

0010342334 *Drawing available*
WPI Acc no: 2000-657564/200064
XRPX Acc No: N2000-487520
Expressing contents of object-oriented database in an intermediate form by deriving intermediate form from textual form expressed according to grammar and configured to populate object-oriented database
Patent Assignee: SUN MICROSYSTEMS INC (SUNM)
Inventor: SAULPAUGH T E; SLAUGHTER G L; TRAVERSAT B A

Patent Family (2 patents, 26 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
EP 1030252	A1	20000823	EP 2000301154	A	20000215	200064	B
US 6611844	B1	20030826	US 1999253868	A	19990219	200357	E

Priority Applications (no., kind, date): US 1999253868 A 19990219

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes
EP 1030252	A1	EN	33	11	
Regional Designated States,Original	AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI				

Alerting Abstract EP A1
NOVELTY - Entries and objects relate to configuration parameters of a computer system while the intermediate form (350) is derived from a textual form expressed according to a grammar (344). The intermediate form is configurable to populate an object-oriented **database** (340). The number of entries is stored in the intermediate form (350) in a computer readable medium.
DESCRIPTION - INDEPENDENT CLAIMS are included for:

A. a **computer program** product comprising **program** instructions for expressing contents of an object oriented **database** in an intermediate form
B. a computer system for expressing contents of an object oriented **database** in an intermediate form

USE - In **computer languages** and object-oriented **database** systems.
ADVANTAGE - Provides an intelligent mechanism and process for storing an object-oriented configuration **database**. No **important** information is **permanently** lost when transforming one into the other, and therefore one

form can be transformed into the other form and back again an indefinite number of times.
DESCRIPTION OF DRAWINGS - The drawing is a block diagram illustrating an overview of the transformation of an object oriented **database** to and from a grammatical form in one embodiment of the invention.
340 object-oriented **database**
344 grammar
350 intermediate form

Title Terms /Index Terms/Additional Words: EXPRESS; CONTENT; OBJECT; ORIENT; **DATABASE**; INTERMEDIATE; FORM; DERIVATIVE; TEXT; ACCORD; GRAMMAR; CONFIGURATION

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-017/30			Main		"Version 7"

US Classification, Issued: 707103000, 707100000, 707102000, 707104100, 717108000, 717116000, 717147000, 717153000, 709316000, 709332000

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): **T01-J05B4C**; **T01-J11A1**; **T01-S01B**; **T01-S03**

Original Publication Data by Authority

EPO

Publication No. EP 1030252 A1 (Update 200064 B)
Publication Date: 20000823
Ein intelligenter Zwischenstatus einer objekt-orientierten Datenbank
An intelligent intermediate state of an object-oriented database
Etat intermediaire et intelligent d'une base de donnees orientees objet
Assignee: SUN MICROSYSTEMS, INC., 901 San Antonio Road, Palo Alto, California 94303, US (SUNM)
Inventor: Saulpaugh, Thomas E., 6938 Bret Harte Drive, San Jose, California 95120, US
Slaughter, Gregory L., 3326 Emerson Street, Palo Alto, California 94306, US
Traversat, Bernard A., 2055 California Street, Apt. 402, San Francisco, California 94109, US
Agent: Harris, Ian Richard, D. Young & Co., 21 New Fetter Lane, London EC4A 1DA, GB
Language: EN (33 pages, 11 drawings)
Application: EP 2000301154 A 20000215 (Local application)
Priority: US 1999253868 A 19990219
Designated States: (Regional Original) AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI
Original IPC: G06F-17/30(A)
Current IPC: G06F-17/30(A)

Original Abstract: A method and system for providing an intelligent intermediate form of an object-oriented database. The intermediate form is derived from a grammatical form of an object-oriented database through the process of compilation. The grammatical form is a persistent form of an object-oriented database expressed in a human-readable and human-editable textual form according to a grammar. The intermediate form comprises an array of intelligent entry objects which encapsulate data with methods for manipulating that data. The methods include creating a database entry, creating a property associated with an entry, creating an attribute associated with an entry or property, querying the last entry, property, or attribute created, and finalizing entry storage. The intermediate form lacks the infrastructure of the database, but the intermediate form can be used to populate the object-oriented database with entries. The object-oriented database is an object-oriented configuration database which stores configuration parameters pertaining to the software and hardware of a computer system, such as application programs, device drivers, system services, and other components. The object-oriented database is platform-independent and is therefore configured to be hosted on several different operating systems and computing platforms.
Claim:

1. A method for expressing contents of an object-oriented database in an intermediate form, said method comprising:
 - expressing a plurality of entries corresponding to objects in an object-oriented database in said intermediate form, wherein said entries and said objects relate to configuration parameters of a computer system, wherein said intermediate form is derived from a textual form expressed according to a grammar, wherein said intermediate form is configurable to populate said object-oriented database;
 - storing said plurality of entries in said intermediate form in a computer readable medium.

United States

Publication No. US 6611844 B1 (Update 200357 E)
Publication Date: 20030826
Method and system for java program storing database object entries in an intermediate form between textual form and an object-oriented form
Assignee: Sun Microsystems, Inc., Santa Clara, CA, US
Inventor: Saulpaugh, Thomas E., San Jose, CA, US
Slaughter, Gregory L., Palo Alto, CA, US
Traversat, Bernard A., San Francisco, CA, US
Agent: Meyertons Hood Kivlin Kowert & Goetzel, P.C., US
Kivlin, B. Noel, US
Language: EN
Application: US 1999253868 A 19990219 (Local application)
Original IPC: G06F-17/30(A)
Current IPC: G06F-17/30(A)
Original US Class (main): 707103
Original US Class (secondary): 707100 707102 707104.1 717108 717116 717147 717153 709316 709332
Original Abstract: A method and system for providing an intelligent intermediate form of an object-oriented database. The intermediate form is derived from a grammatical form of an object-oriented database through the process of compilation. The grammatical form is a persistent form of an object-oriented database expressed in a human-readable and human-editable textual form according to a grammar. The intermediate form comprises an

array of intelligent entry objects which encapsulate data with methods for manipulating that data. The methods include creating a database entry, creating a property associated with an entry, creating an attribute associated with an entry or property, querying the last entry, property, or attribute created, and finalizing entry storage. The intermediate form lacks the infrastructure of the database, but the intermediate form can be used to populate the object-oriented database with entries. The object-oriented database is an object-oriented configuration database which stores configuration parameters pertaining to the software and hardware of a computer system, such as application programs, device drivers, system services, and other components. The object-oriented database is platform-independent and is therefore configured to be hosted on several different operating systems and computing platforms.

Claim: What is claimed is:

1. 1. A method for expressing contents of an object-oriented database in an intermediate form for sending to a Java program, said method comprising:
 - expressing a plurality of entries corresponding to objects in said object-oriented database in said intermediate form, wherein said entries and said objects relate to configuration parameters of a computer system, wherein said intermediate form is derived from a textual form expressed according to a grammar, wherein said object-oriented database stores configuration data that identifies available resources of said computer system, and wherein said intermediate form is configurable to populate said object-oriented database;
 - sending said plurality of entries in said intermediate form to said Java program; and
 - said Java program storing said plurality of entries in said intermediate form in a computer readable medium for populating another object-oriented database.

US006611844B1

(12) **United States Patent**
Saulpaugh et al.

(10) **Patent No.: US 6,611,844 B1**
(45) **Date of Patent: *Aug. 26, 2003**

(54) **METHOD AND SYSTEM FOR JAVA PROGRAM STORING DATABASE OBJECT ENTRIES IN AN INTERMEDIATE FORM BETWEEN TEXTUAL FORM AND AN OBJECT-ORIENTED FORM**

(75) Inventors: **Thomas E. Saulpaugh**, San Jose, CA (US); **Gregory L. Slaughter**, Palo Alto, CA (US); **Bernard A. Traversat**, San Francisco, CA (US)

(73) Assignee: **Sun Microsystems, Inc.**, Santa Clara, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **09/253,868**

(22) Filed: **Feb. 19, 1999**

(51) **Int. Cl.**[7] **G06F 17/30**

(52) **U.S. Cl.** **707/103**; 707/100; 707/102; 707/104.1; 717/108; 717/116; 717/147; 717/153; 709/316; 709/332

(58) **Field of Search** 707/11, 100–104.1, 707/200; 709/201, 223, 230, 310, 316, 332; 717/1–11, 108, 116, 136, 140–148, 150–153, 118–119, 166–165

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,335,346 A * 8/1994 Fabbio 711/163

(List continued on next page.)

FOREIGN PATENT DOCUMENTS

EP 0 256 881 2/1988

OTHER PUBLICATIONS

Ege, R.K. et al., A modular Java API for object–oriented databases, computer software and applications conference, 1998, COMPSAC '98, Aug. 1998, pp. 55–60.*

Tennent, *Principles of Programming Languages*, Prentice Hall 1981, pp. 9–33.

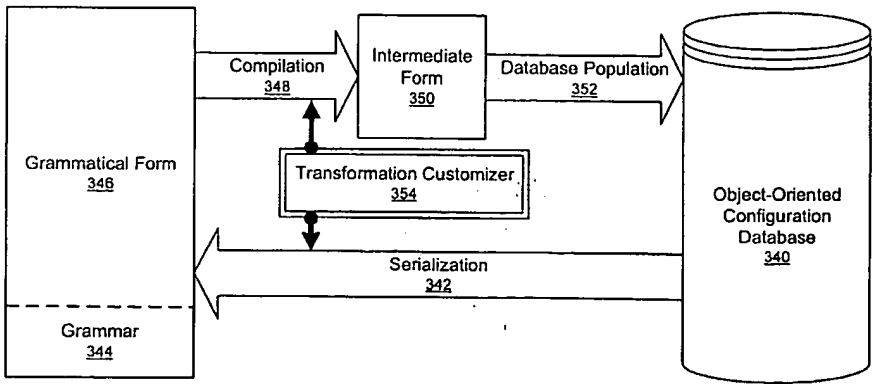
Majka, "Getting Acquainted with NetInfo," *NEXTSTEP In Focus*, Summer 1993, vol. 3, Issue 3, copyright NeXT Computer, Inc. 1993. 10 pages.

Primary Examiner—Srirama Channavajjala
(74) *Attorney, Agent, or Firm*—Meyertons Hood Kivlin Kowert & Goetzel, P.C.; B. Noël Kivlin

(57) **ABSTRACT**

A method and system for providing an intelligent intermediate form of an object-oriented database. The intermediate form is derived from a grammatical form of an object-oriented database through the process of compilation. The grammatical form is a persistent form of an object-oriented database expressed in a human-readable and human-editable textual form according to a grammar. The intermediate form comprises an array of intelligent entry objects which encapsulate data with methods for manipulating that data. The methods include creating a database entry, creating a property associated with an entry, creating an attribute associated with an entry or property, querying the last entry, property, or attribute created, and finalizing entry storage. The intermediate form lacks the infrastructure of the database, but the intermediate form can be used to populate the object-oriented database with entries. The object-oriented database is an object-oriented configuration database which stores configuration parameters pertaining to the software and hardware of a computer system, such as application programs, device drivers, system services, and other components. The object-oriented database is platform-independent and is therefore configured to be hosted on several different operating systems and computing platforms.

33 Claims, 11 Drawing Sheets



then, the StaticTreeSerializer and StaticTreeCompiler classes can only read and write values in terms of the following primitive data types: string, boolean, int, long, short, byte, char, double, float, and arrays thereof. However, some object-oriented databases may define complex data types: that is, data types which combine one or more of the primitive data types. For example, a complex data type could be an object which includes a string, two long integers, and an array of boolean values. To modify the grammar to describe these complex data types would render the grammar large and unwieldy. Therefore, to enable serialization and compilation to understand complex data types, in one embodiment a customization plug-in is provided.

In one embodiment, the plug-in comprises a Java™ Language interface called StaticTreePlugIn, which is defined as follows:

```
package javaos.javax.system.database.statictree;
import java.io.*;
import java.util.Hashtable;
public interface StaticTreePlugIn {
    public String compile(Hashtable properties) throws
        IllegalArgumentException;
    public void serialize(Hashtable properties) throws
        IllegalArgumentException;
}
```

The StaticTreePlugIn interface should be implemented by a plug-in class for utilizing complex data types. In other words, developers who wish to utilize complex data types should design a particular plug-in class that implements the StaticTreePlugIn interface. In one embodiment, an example of such a plug-in is the "business card" discussed in conjunction with the database description grammar. The business card is a complex data type that includes a collection of properties and/or attributes for a particular software component.

In one embodiment, a plug-in functions as follows. When compiling a grammatical form that contains a complex data type, the compiler determines that the data type is not a known or recognized data type. The compiler then creates a hash table containing the primitive data types which make up the complex data type. In other words, the hash table contains the properties of the complex data type. The compiler then calls the compile() method of the plug-in and passes the hash table to the plug-in. The plug-in collapses the data types in the hash table down to a single, complex object. The compile() method of the plug-in returns the name of the complex property in String format. When serializing, on the other hand, the serializer passes a blank hash table to the serialize() method of the plug-in. The serialize() method of the plug-in should translate the complex object into a series of primitives and fill the hash table with the list of primitives.

Various embodiments of the present invention further include receiving or storing instructions and/or data implemented in accordance with the foregoing description upon a carrier medium. Suitable carrier mediums include storage mediums such as disk, as well as electrical signals or digital signals conveyed via a communication medium such as network **108** or a wireless link.

While the present invention has been described with reference to particular embodiments, it will be understood that the embodiments are illustrated and that the invention scope is not so limited. Any variations, modifications, additions and improvements to the embodiments described are possible. These variations, modifications, additions and improvements may fall within the scope of the invention as detailed within the following claims.

What is claimed is:

1. A method for expressing contents of an object-oriented database in an intermediate form for sending to a Java program, said method comprising:

expressing a plurality of entries corresponding to objects in said object-oriented database in said intermediate form, wherein said entries and said objects relate to configuration parameters of a computer system, wherein said intermediate form is derived from a textual form expressed according to a grammar, wherein said object-oriented database stores configuration data that identifies available resources of said computer system, and wherein said intermediate form is configurable to populate said object-oriented database;

sending said plurality of entries in said intermediate form to said Java program; and

said Java program storing said plurality of entries in said intermediate form in a computer readable medium for populating another object-oriented database.

2. The method of claim **1**,

wherein said object-oriented database is configured to be platform independent.

3. The method of claim **1**,

wherein said contents of said object-oriented database pertain to one or more application programs.

4. The method of claim **1**,

wherein said textual form include name-value pairs corresponding to properties and attributes of software and hardware of said computer system.

5. The method of claim **1**,

wherein said textual form expresses a hierarchy of entries.

6. The method of claim **1**,

wherein said intermediate form includes an array of entries configurable to populate said object-oriented database.

7. The method of claim **1**,

wherein said entries of said intermediate form are objects, wherein said objects encapsulate data with methods.

8. The method of claim **1**,

wherein said intermediate form is configurable to populate said object-oriented database with said entries through a programming interface for accessing said object-oriented database.

9. The method of claim **1**, wherein said configuration data identifies physical devices present in said computer system.

10. The method of claim **1**, wherein said configuration data identifies software services installed in said computer system.

11. A carrier medium comprising program instructions for expressing contents of an object-oriented database in an intermediate form for sending to a Java program, wherein said program instructions are executable to implement:

expressing a plurality of entries corresponding to objects in said object-oriented database in said intermediate form, wherein said entries and said objects relate to configuration parameters of a computer system, wherein said intermediate form is derived from a textual form expressed according to a grammar, wherein said object-oriented database stores configuration data that identifies available resources of said computer system, and wherein said intermediate form is configurable to populate said object-oriented database;

sending said plurality of entries in said intermediate form to said Java program; and

said Java program storing said plurality of entries in said intermediate form in a computer readable medium for populating another object-oriented database.

12. The carrier medium of claim 11,

wherein said object-oriented database is configured to be platform independent.

13. The carrier medium of claim 9,

wherein said contents of said object-oriented database pertain to one or more application programs.

14. The carrier medium of claim 11,

wherein said textual form include name-value pairs corresponding to properties and attributes of software and hardware of said computer system.

15. The carrier medium of claim 11,

wherein said textual form expresses a hierarchy of entries.

16. The carrier medium of claim 11,

wherein said intermediate form includes an array of entries configurable to populate said object-oriented database.

17. The carrier medium of claim 11,

wherein said entries of said intermediate form are objects, wherein said objects encapsulate data with methods.

18. The carrier medium of claim 11,

wherein said intermediate form is configurable to populate said object-oriented database with said entries through a programming interface for accessing said object-oriented database.

19. The carrier medium of claim 11,

wherein said carrier medium is a storage medium.

20. The carrier medium of claim 11, wherein said configuration data identifies physical devices present in said computer system.

21. The carrier medium of claim 11, wherein said configuration data identifies software services installed in said computer system.

22. A computer system for expressing contents of an object-oriented database in an intermediate form for sending to a Java program, said computer system comprising:

a CPU; and

a memory coupled to said CPU;

wherein said memory stores said intermediate form and said object-oriented database;

wherein said memory stores program instructions executable by said CPU, wherein said program instructions are executable to:

express a plurality of entries corresponding to objects in said object-oriented database in said intermediate form, wherein said entries and said objects relate to configuration parameters of said computer system, wherein said intermediate form is derived from a textual form expressed according to a grammar, wherein said object-oriented database stores configuration data that identifies available resources of said computer system, and wherein said intermediate form is configurable to populate said object-oriented database;

sending said plurality of entries in said intermediate form to said Java program; and

said Java program storing said plurality of entries in said intermediate form in a computer readable-medium for populating another object-oriented database.

23. The computer system of claim 22,

wherein said object-oriented database is configured to be platform independent.

24. The computer system of claim 22,

wherein said contents of said object-oriented database pertain to one or more application programs.

25. The computer system of claim 22,

wherein said textual form include name-value pairs corresponding to properties and attributes of software and hardware of said computer.

26. The computer system of claim 22,

wherein said textual form expresses a hierarchy of entries.

27. The computer system of claim 22,

wherein said intermediate form includes an array of entries configurable to populate said object-oriented database.

28. The computer system of claim 22,

wherein said entries of said intermediate form are objects, wherein said objects encapsulate data with methods.

29. The computer system of claim 22,

wherein said intermediate form is configurable to populate said object-oriented database with said entries through a programming interface for accessing said object-oriented database.

30. The computer system of claim 22, wherein said configuration data identifies physical devices present in said computer system.

31. The computer system of claim 22, wherein said configuration data identifies software services installed in said computer system.

32. A method, comprising:

a Java program on a computer system storing configuration data identifying available resources of a plurality of computer systems in an object-oriented database, wherein the object-oriented database includes a plurality of objects related to configuration parameters of each of the plurality of computer systems;

the Java program receiving an event notification indicating another Java program on another computer system of the plurality of computer systems, wherein the event notification relates to at least part of the configuration data in the object-oriented database;

the Java program expressing one or more of the plurality of objects in said object-oriented database corresponding to the event notification into a textual form according to a grammar;

the Java program sending the textual form to the another Java program on the another computer system;

the another Java program deriving a plurality of entries in an intermediate form from the textual form expressed according to the grammar; and

the another Java program populating another object-oriented database of the another computer system from the intermediate form.

33. The method as recited in claim 32, wherein said expressing includes the Java program extending the grammar, wherein said sending includes sending the extended grammar to the another Java program, and wherein said deriving includes the another Java program deriving the plurality of entries according to the extended grammar.

* * * * *

65/7/47 (Item 47 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

0010342333 *Drawing available*
WPI Acc no: 2000-657563/200064
XRPX Acc No: N2000-487519
Transforming contents of object-oriented database into intermediate object-oriented form by expressing several entries corresponding to objects in object-oriented database in intermediate form
Patent Assignee: SUN MICROSYSTEMS INC (SUNM)
Inventor: SAULPAUGH T E; SLAUGHTER G L; TRAVERSAT B A

Patent Family (2 patents, 26 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
EP 1030251	A1	20000823	EP 2000301152	A	20000215	200064	B
US 6598052	B1	20030722	US 1999253841	A	19990219	200354	E

Priority Applications (no., kind, date): US 1999253841 A 19990219

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes
EP 1030251	A1	EN	33	11	
Regional Designated States,Original	AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI				

Alerting Abstract EP A1
NOVELTY - A textual form is parsed into a series of tokens that are compiled (348) into a number of entries expressed the number of entries corresponding to the objects in the object-oriented **database** in an intermediate form (350). The latter may be configured to populate an object-oriented **database** (340).
DESCRIPTION - INDEPENDENT CLAIMS are included for:

A. a **computer program** product comprising **program** instructions for transforming an object oriented **database** expressed in textual form according to grammar into an intermediate object oriented form
B. a computer system for transforming an object oriented **database** expressed in textual form according to grammar into an intermediate object oriented form

USE - In **computer languages** and object-oriented **database** systems.
ADVANTAGE - Provides an intelligent mechanism and process for storing an object-oriented configuration **database**. No **important** information is **permanently** lost when transforming one into the other, and therefore one

form can be transformed into the other form and back again an indefinite number of times.
DESCRIPTION OF DRAWINGS - The drawing
340 object-oriented **database**
348 compiled
350 intermediate form

Title Terms /Index Terms/Additional Words: TRANSFORM; CONTENT; OBJECT; ORIENT; **DATABASE**; INTERMEDIATE; FORM; EXPRESS; ENTER; CORRESPOND

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-017/30			Main		"Version 7"

US Classification, Issued: 707102000, 707100000, 707103000, 707104000, 717108000, 717116000, 717147000, 717165000, 717152000

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): **T01-J05B4C**; **T01-J11A1**; **T01-S01B**; **T01-S03**

Original Publication Data by Authority

EPO

Publication No. EP 1030251 A1 (Update 200064 B)
Publication Date: 20000823
Verfahren um die Grammatik einer Konfigurationsdatenbank in eine Zwischenform umzuwandeln, die die Erzeugung einer Datenbank vereinfacht
Process for transforming configuration database grammar into intermediate form that simplifies database generation
Procede servant a transformer une base de donnees de configuration en format intermediaire pour simplifier la generation d'une base de donnees
Assignee: SUN MICROSYSTEMS, INC., 901 San Antonio Road, Palo Alto, California 94303, US (SUNM)
Inventor: Saulpaugh, Thomas E., 6938 Bret Harte Drive, San Jose, California 95120, US
Slaughter, Gregory L., 3326 Emerson Street, Palo Alto, California 94306, US
Traversat, Bernard A., 2055 California Street, Apt. 402, San Francisco, California 94109, US
Agent: Harris, Ian Richard, D. Young & Co., 21 New Fetter Lane, London EC4A 1DA, GB
Language: EN (33 pages, 11 drawings)
Application: EP 2000301152 A 20000215 (Local application)
Priority: US 1999253841 A 19990219
Designated States: (Regional Original) AL AT BE CH CY DE DK ES FI FR GB GR IE IT LI LT LU LV MC MK NL PT RO SE SI

Original IPC: G06F-17/30(A)
Current IPC: G06F-17/30(A)
Original Abstract: A method and system for compiling a grammatical form of an object-oriented database into an intermediate form of that database. The grammatical form is a persistent form of an object-oriented database expressed in a human-readable and human-editable textual form according to a grammar. The textual form is parsed into a series of tokens. The tokens are compiled into a plurality of entries. The plurality of entries are expressed in an intermediate form. The intermediate form comprises an array of intelligent entry objects which encapsulate data with methods for manipulating that data. The methods include creating a database entry, creating a property associated with an entry, creating an attribute associated with an entry or property, querying the last entry, property, or attribute created, and finalizing entry storage. The intermediate form lacks the infrastructure of the database, but the intermediate form can be used to populate the object-oriented database with entries. The object-oriented database is an object-oriented configuration database which stores configuration parameters pertaining to the software and hardware of a computer system, such as application programs, device drivers, system services, and other components. The object-oriented database is platform-independent and is therefore configured to be hosted on several different operating systems and computing platforms.
Claim:

1. A method for transforming contents of an object-oriented database expressed in a textual form according to a grammar into an intermediate object-oriented form, said textual form and said intermediate form being stored in a memory of a computer system, said method comprising:
 - expressing a plurality of entries corresponding to objects in said object-oriented database in said textual form according to said grammar, wherein said entries and said objects relate to configuration parameters of software and hardware of said computer system;
 - parsing said textual form into a series of tokens;
 - compiling said series of tokens into said plurality of entries;
 - expressing said plurality of entries corresponding to said objects in said object-oriented database in an intermediate form, wherein said intermediate form is configurable to populate said object-oriented database.

United States

Publication No. US 6598052 B1 (Update 200354 E)
Publication Date: 20030722
Method and system for transforming a textual form of object-oriented database entries into an intermediate form configurable to populate an object-oriented database for sending to java program
Assignee: Sun Microsystems, Inc., Santa Clara, CA, US
Inventor: Saulpaugh, Thomas E., San Jose, CA, US
Slaughter, Gregory L., Palo Alto, CA, US
Traversat, Bernard A., San Francisco, CA, US
Agent: Meyertons Hood Kivlin Kowert & Goetzel, P.C., US
Kivlin, B. Noel, US
Language: EN
Application: US 1999253841 A 19990219 (Local application)
Original IPC: G06F-17/30(A)
Current IPC: G06F-17/30(A)
Original US Class (main): 707102

Original US Class (secondary): 707100 707103 707104 717108 717116 717147 717165 717152
Original Abstract: A method and system for compiling a grammatical form of an object-oriented database into an intermediate form of that database. The grammatical form is a persistent form of an object-oriented database expressed in a human-readable and human-editable textual form according to a grammar. The textual form is parsed into a series of tokens. The tokens are compiled into a plurality of entries. The plurality of entries are expressed in an intermediate form. The intermediate form comprises an array of intelligent entry objects which encapsulate data with methods for manipulating that data. The methods include creating a database entry, creating a property associated with an entry, creating an attribute associated with an entry or property, querying the last entry, property, or attribute created, and finalizing entry storage. The intermediate form lacks the infrastructure of the database, but the intermediate form can be used to populate the object-oriented database with entries. The object-oriented database is an object-oriented configuration database which stores configuration parameters pertaining to the software and hardware of a computer system, such as application programs, device drivers, system services, and other components. The object-oriented database is platform-independent and is therefore configured to be hosted on several different operating systems and computing platforms.
Claim: What is claimed is:

1. 1. A method for transforming contents of an object-oriented database expressed in a textual form according to a grammar into an intermediate object-oriented form for sending to a Java program, said textual form and said intermediate form being stored in a memory of a computer system, said method comprising:
 - expressing a plurality of entries corresponding to objects in said object-oriented database in said textual form according to said grammar, wherein said entries and said objects relate to configuration parameters of said computer system;
 - parsing said textual form into a series of tokens;
 - compiling said series of tokens into said plurality of entries;
 - expressing said plurality of entries corresponding to said objects in said object-oriented database in an intermediate form, wherein said intermediate form is configurable to populate said object-oriented database, and wherein said object-oriented database stores configuration data that identifies available resources of said computer system; and
 - sending said intermediate form to said Java program for populating another object-oriented database.

US006598052B1

(12) **United States Patent**
Saulpaugh et al.

(10) **Patent No.: US 6,598,052 B1**
(45) **Date of Patent: *Jul. 22, 2003**

(54) **METHOD AND SYSTEM FOR TRANSFORMING A TEXTUAL FORM OF OBJECT-ORIENTED DATABASE ENTRIES INTO AN INTERMEDIATE FORM CONFIGURABLE TO POPULATE AN OBJECT-ORIENTED DATABASE FOR SENDING TO JAVA PROGRAM**

(75) Inventors: **Thomas E. Saulpaugh**, San Jose, CA (US); **Gregory L. Slaughter**, Palo Alto, CA (US); **Bernard A. Traversat**, San Francisco, CA (US)

(73) Assignee: **Sun Microsystems, Inc.**, Santa Clara, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

This patent is subject to a terminal disclaimer.

(21) Appl. No.: **09/253,841**

(22) Filed: **Feb. 19, 1999**

(51) **Int. Cl.**[7] **G06F 17/30**

(52) **U.S. Cl.** **707/102**; 707/100; 707/103; 707/104; 717/108; 717/116; 717/147; 717/165; 717/152

(58) **Field of Search** 707/100–104.1, 707/1, 200; 709/201, 223, 230, 310; 717/1–11, 108, 116, 136, 140, 144–148, 152–153, 164

(56) **References Cited**

U.S. PATENT DOCUMENTS

5,335,346 A * 8/1994 Fabbio 711/163
5,694,598 A * 12/1997 Durand et al. 707/103

(List continued on next page.)

FOREIGN PATENT DOCUMENTS

EP 0 256 881 2/1988
EP 0 -256881 * 2/1998 15/40

OTHER PUBLICATIONS

Ege, R.K. Storing Java objects in any database, IEEE technology of object–oriented languages and systems, 1999 tools 30 proceedings, Aug., 1999, pp.: 312–321.*

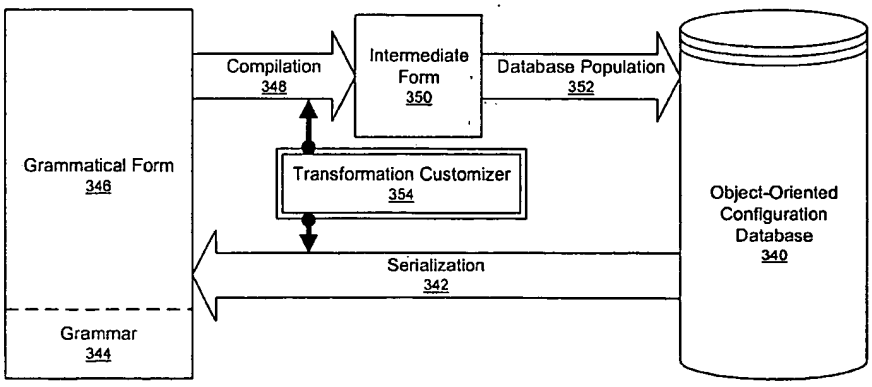
(List continued on next page.)

Primary Examiner—Srirama Channavajjala
(74) *Attorney, Agent, or Firm*—Meyertons Hood Kivlin Kowert & Goetzel, P.C.; B. Noël Kivlin

(57) **ABSTRACT**

A method and system for compiling a grammatical form of an object-oriented database into an intermediate form of that database. The grammatical form is a persistent form of an object-oriented database expressed in a human-readable and human-editable textual form according to a grammar. The textual form is parsed into a series of tokens. The tokens are compiled into a plurality of entries. The plurality of entries are expressed in an intermediate form. The intermediate form comprises an array of intelligent entry objects which encapsulate data with methods for manipulating that data. The methods include creating a database entry, creating a property associated with an entry, creating an attribute associated with an entry or property, querying the last entry, property, or attribute created, and finalizing entry storage. The intermediate form lacks the infrastructure of the database, but the intermediate form can be used to populate the object-oriented database with entries. The object-oriented database is an object-oriented configuration database which stores configuration parameters pertaining to the software and hardware of a computer system, such as application programs, device drivers, system services, and other components. The object-oriented database is platform-independent and is therefore configured to be hosted on several different operating systems and computing platforms.

33 Claims, 11 Drawing Sheets



database description grammar. Like all JSD populators, however, grammar-based populators must implement the TreePopulator interface. The StaticTreePopulator class is the default implementation of a grammar-based populator. The StaticTreePopulator class encapsulates a compiler (StaticTreeCompiler) and a compiled tree of grammatically derived entries. As described above, the compilation takes place during the StaticTreePopulator construction process. If the compilation completes correctly, then the StaticTreePopulator is ready to populate the JSD.

The StaticTreePopulator understands the intermediate database format. Each invocation of the TreePopulator interface requires an intermediate-form entry lookup, and possibly a property or attribute lookup as well The results of the lookup (entries, properties, or attributes) are then returned to the JSD for inclusion in the active cache of entries. During population, each level of the tree is read an entry at a time until all entries are returned to the JSD by the populator for inclusion into the active cache of published JSD entries. Each entry returned to the JSD is identified by a integer, known as an entry ID. The entry IDs are returned by the populator to the JSD. The JSD observes the integer as an opaque token unique to a single entry within the populator.

The first entry ID is used to kick-start the population process and is returned to the JSD by the getRootEntry() method. All subsequent entry IDs are returned by the following methods: getParentEntry(), getFirstChildEntry(), and getPeerEntry(). Each property associated with an entry is identified by a combination of an entry ID and a property name. Similarly, each attribute associated with an entry is identified by a combination of an entry ID and a attribute name. Each attribute associated with a property is identified by the tuple of entry ID, property name, and attribute name.

Normally, serialization and compilation as described above are only able to translate a finite, pre-defined set of primitive Java™ Language data types. In one embodiment, then, the StaticTreeSerializer and StaticTreeCompiler classes can only read and write values in terms of the following primitive data types: string, boolean, int, long, short, byte, char, double, float, and arrays thereof. However, some object-oriented databases may define complex data types: that is, data types which combine one or more of the primitive data types. For example, a complex data type could be an object which includes a string, two long integers, and an array of boolean values. To modify the grammar to describe these complex data types would render the grammar large and unwieldy. Therefore, to enable serialization and compilation to understand complex data types, in one embodiment a customization plug-in is provided.

In one embodiment, the plug-in comprises a Java™ Language interface called StaticTreePlugIn, which is defined as follows:

```
package javaos.javax.system.database.statictree;
import java.io.*;
import java.util.Hashtable;
public interface StaticTreePlugIn {
    public String compile(Hashtable properties) throws
        IllegalArgumentException;
    public void serialize(Hashtable properties) throws
        IllegalArgumentException;
}
```

The StaticTreePlugIn interface should be implemented by a plug-in class for utilizing complex data types. In other words, developers who wish to utilize complex data types should design a particular plug-in class that implements the StaticTreePlugIn interface. In one embodiment, an example of such a plug-in is the "business card" discussed in conjunction with the database description grammar. The business card is a complex data type that includes a collection of properties and/or attributes for a particular software component.

In one embodiment, a plug-in functions as follows. When compiling a grammatical form that contains a complex data type, the compiler determines that the data type is not a known or recognized data type. The compiler then creates a hash table containing the primitive data types which make up the complex data type. In other words, the hash table contains the properties of the complex data type. The compiler then calls the compile() method of the plug-in and passes the hash table to the plug-in. The plug-in collapses the data types in the hash table down to a single, complex object. The compile() method of the plug-in returns the name of the complex property in String format. When serializing, on the other hand, the serializer passes a blank hash table to the serialize() method of the plug-in. The serialize() method of the plug-in should translate the complex object into a series of primitives and fill the hash table with the list of primitives.

Various embodiments of the present invention further include receiving or storing instructions and/or data implemented in accordance with the foregoing description upon a carrier medium. Suitable carrier mediums include storage mediums such as disk, as well as electrical signals or digital signals conveyed via a communication medium such as network **108** or a wireless link.

While the present invention has been described with reference to particular embodiments, it will be understood that the embodiments are illustrated and that the invention scope is not so limited. Any variations, modifications, additions and improvements to the embodiments described are possible. These variations, modifications, additions and improvements may fall within the scope of the invention as detailed within the following claims.

What is claimed is:

1. A method for transforming contents of an object-oriented database expressed in a textual form according to a grammar into an intermediate object-oriented form for sending to a Java program, said textual form and said intermediate form being stored in a memory of a computer system, said method comprising:

expressing a plurality of entries corresponding to objects in said object-oriented database in said textual form according to said grammar, wherein said entries and said objects relate to configuration parameters of said computer system;

parsing said textual form into a series of tokens;

compiling said series of tokens into said plurality of entries;

expressing said plurality of entries corresponding to said objects in said object-oriented database in an intermediate form, wherein said intermediate form is configurable to populate said object-oriented database, and wherein said object-oriented database stores configuration data that identifies available resources of said computer system; and

sending said intermediate form to said Java program for populating another object-oriented database.

2. The method of claim **1**,

wherein said object-oriented database is configured to be platform independent.

3. The method of claim **1**,

wherein said contents of said object-oriented database pertain to one or more application programs.

4. The method of claim 1,

wherein said tokens include keywords specified by said grammar.

5. The method of claim 4,

wherein said keywords are selected from the group consisting of: tree, entry, properties, and attributes.

6. The method of claim 1,

wherein said tokens include name-value pairs corresponding to properties and attributes of said software and hardware of said computer system.

7. The method of claim 1,

wherein said textual form expresses a hierarchy of entries;

wherein said parsing said textual form into a series of tokens occurs recursively to parse said hierarchy of entries.

8. The method of claim 1,

wherein said intermediate form includes an array of entries configurable to populate said object-oriented database.

9. The method of claim 1,

wherein said entries of said intermediate form are objects, wherein said objects encapsulate data with methods.

10. The method of claim 1, wherein said configuration data identifies physical devices present in said computer system.

11. The method of claim 1, wherein said configuration data identifies software services installed in said computer system.

12. A carrier medium comprising program instructions for transforming contents of an object-oriented database expressed in a textual form according to a grammar into an intermediate object-oriented form for sending to a Java program, wherein said program instructions are executable to implement:

expressing a plurality of entries corresponding to objects in said object-oriented database in said textual form according to said grammar, wherein said entries and said objects relate to configuration parameters of a computer system;

parsing said textual form into a series of tokens;

compiling said series of tokens into said plurality of entries; and

expressing said plurality of entries corresponding to said objects in said object-oriented database in an intermediate form, wherein said intermediate form is configurable to populate said object-oriented database, and wherein said object-oriented database stores configuration data that identifies available resources of said computer system; and

sending said intermediate form to said Java program for populating another object-oriented database.

13. The carrier medium of claim 12,

wherein said object-oriented database is configured to be platform independent.

14. The carrier medium of claim 12,

wherein said contents of said object-oriented database pertain to one or more application programs.

15. The carrier medium of claim 12,

wherein said tokens include keywords specified by said grammar.

16. The carrier medium of claim 15,

wherein said keywords are selected from the group consisting of: tree, entry, properties, and attributes.

17. The carrier medium of claim 12,

wherein said tokens include name-value pairs corresponding to properties and attributes of said software and hardware of said computer system.

18. The carrier medium of claim 12,

wherein said textual form expresses a hierarchy of entries;

wherein said parsing said textual form into a series of tokens occurs recursively to parse said hierarchy of entries.

19. The carrier medium of claim 12,

wherein said intermediate form includes an array of entries configurable to populate said object-oriented database.

20. The carrier medium of claim 12,

wherein said entries of said intermediate form are objects, wherein said objects encapsulate data with methods.

21. The carrier medium of claim 12,

wherein said carrier medium is a storage medium.

22. The carrier medium of claim 12, wherein said configuration data identifies physical devices present in said computer system.

23. The carrier medium of claim 12, wherein said configuration data identifies software services installed in said computer system.

24. A computer system for transforming contents of an object-oriented database expressed in a textual form according to a grammar into an intermediate object-oriented form for sending to a Java program, said computer system comprising:

a CPU;

a memory coupled to said CPU;

wherein said memory stores said textual form, said intermediate form, and said object-oriented database;

wherein said memory stores program instructions executable by said CPU, wherein said program instructions are executable to:

express a plurality of entries corresponding to objects in said object-oriented database in said textual form according to said grammar, wherein said entries and said objects relate to configuration parameters of a computer;

parse said textual form into a series of tokens;

compile said series of tokens into said plurality of entries;

express said plurality of entries corresponding to said objects in said object-oriented database in an intermediate form, wherein said intermediate form is configurable to populate said object-oriented database, and wherein said object-oriented database stores configuration data that identifies available resources of said computer system; and

sending said intermediate form to said Java program for populating another object-oriented database.

25. The computer system of claim 24,

wherein said object-oriented database is configured to be platform independent.

26. The computer system of claim 24,

wherein said contents of said object-oriented database pertain to one or more application programs.

27. The computer system of claim 24,

wherein said tokens include keywords specified by said grammar.

28. The computer system of claim 24,

wherein said tokens include name-value pairs corresponding to properties and attributes of said software and hardware of said computer.

29. The computer system of claim 24,

wherein said textual form expresses a hierarchy of entries;

wherein in parsing said textual form into a series of tokens, said program instructions are further executable to parse said textual form recursively to parse said hierarchy of entries.

30. The computer system of claim 24,

wherein said intermediate form includes an array of entries configurable to populate said object-oriented database.

31. The computer system of claim 24,

wherein said entries of said intermediate form are objects, wherein said objects encapsulate data with methods.

32. The computer system of claim 24, wherein said configuration data identifies physical devices present in said computer system.

33. The computer system of claim 24, wherein said configuration data identifies software services installed in said computer system.

* * * * *

65/7/40 (Item 40 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

0010901367 *Drawing available*
WPI Acc no: 2001-522164/200157
XRPX Acc No: N2001-386995
Managing information in an information system having a server, a client and database for computers, telecommunications and computer network systems, particularly over a network
Patent Assignee: LIPKIN D S (LIPK-I); SABA SOFTWARE INC (SABA-N)
Inventor: HELGESON C S; LARSON R S; LIPKIN D S; PANUGANTI S

Patent Family (6 patents, 92 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
WO 2001052118	A2	20010719	WO 2001US920	A	20010112	200157	B
AU 200129371	A	20010724	AU 200129371	A	20010112	200166	E
US 20020073080	A1	20020613	US 2000176137	P	20000114	200243	E
			US 2001760432	A	20010112		
US 6643652	B2	20031104	US 2000176137	P	20000114	200374	E
			US 2001759678	A	20010112		
US 6721747	B2	20040413	US 2000176137	P	20000114	200425	E
			US 2001760432	A	20010112		
AU 2001229371	A8	20051020	AU 2001229371	A	20010112	200615	E

Priority Applications (no., kind, date): US 2001760432 A 20010112; US 2001759678 A 20010112; US 2000176137 P 20000114

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
WO 2001052118	A2	EN	233	17		
National Designated States,Original	AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CR CU CZ DE DK DM DZ EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM TR TT TZ UA UG US UZ VN YU ZA ZW					
Regional Designated States,Original	AT BE CH CY DE DK EA ES FI FR GB GH GM GR IE IT KE LS LU MC MW MZ NL OA PT SD SE SL SZ TR TZ UG ZW					
AU 200129371	A	EN			Based on OPI patent	WO 2001052118
US 20020073080	A1	EN			Related to Provisional	US 2000176137
US 6643652	B2	EN			Related to Provisional	US 2000176137

US 6721747	B2	EN			Related to Provisional	US 2000176137
AU 2001229371	A8	EN			Based on OPI patent	WO 2001052118

Alerting Abstract WO A2

NOVELTY - Method entails **generating metadata** using an **import** agent, determining at least one match using a match agent, and **dispatching** the result associated with the match using a delivery agent. The **metadata** may be RDF **metadata**, or the match agent may use an RQL query. A match template may be used in the information resource system having at least two sets of **metadata**; or an **import** agent, match agent and **metadata** repository.
DESCRIPTION - AN INDEPENDENT CLAIM is made for:

A. An article of manufacture; and
B. A **computer program** product for use with the information resource system.

USE - For computers, telecommunications, and computer network systems, particularly for efficiently managing and retrieving information over a network.
ADVANTAGE - Enables searching for information and discovering information, such as web resources, in a more flexible and sophisticated manner. Invention also facilitates finding information associated with RDF on the World-Wide Web.
DESCRIPTION OF DRAWINGS - Drawing shows a typical configuration of Internet **connected** systems representative of preferred embodiment of the present invention.

Title Terms /Index Terms/Additional Words: MANAGE; INFORMATION; SYSTEM; SERVE; CLIENT; **DATABASE**; COMPUTER; TELECOMMUNICATION; NETWORK

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-017/30; G06F-007/00			Main		"Version 7"

US Classification, Issued: 707003000, 707010000, 707104100, 709202000, 709203000, 707010000, 707003000, 707008000, 707100000, 707200000, 709200000 , 709202000, 709217000, 709225000, 715501100, 715513000, 715523000

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): **T01-H07C5A**; **T01-H07C5E**; **T01-H07C5S**; **T01-J05A2**; **T01-J05B2**; **T01-J05B3**; **T01-S03**

Original Publication Data by Authority

Australia

Publication No. AU 2001229371 A8 (Update 200615 E)
Publication Date: 20051020
Information server
Assignee: SABA SOFTWARE INC (SABA-N)
Inventor: LIPKIN D S
Language: EN
Application: AU 2001229371 A 20010112 (Local application)
Priority: US 2000176137 P 20000114
Related Publication: WO 2001052118 A (Based on OPI patent)
Original IPC: G06F-17/30(A)
Current IPC: G06F-17/30(A)

Publication No. AU 200129371 A (Update 200166 E)
Publication Date: 20010724
Assignee: SABA SOFTWARE INC; US (SABA-N)
Language: EN
Application: AU 200129371 A 20010112 (Local application)
Priority: US 2000176137 P 20000114
Related Publication: WO 2001052118 A (Based on OPI patent)

United States

Publication No. US 20020073080 A1 (Update 200243 E)
Publication Date: 20020613
Method and apparatus for an information server
Assignee: Lipkin, Daniel S., Belmont, CA, US (LIPK-I)
Inventor: Lipkin, Daniel S., Belmont, CA, US
Agent: MORRISON & FOERSTER LLP, 425 MARKET STREET, SAN FRANCISCO, CA, US
Language: EN
Application: US 2000176137 P 20000114 (Related to Provisional)
US 2001760432 A 20010112 (Local application)
Original IPC: G06F-7/00(A)
Current IPC: G06F-7/00(A)
Original US Class (secondary): 7073
Original Abstract: The present invention provides a method and apparatus for managing information in an information resource system containing a server, a client, and a database, by generating metadata using an import agent, determining at least one match using a match agent, and dispatching the at least one match or a result associated with the match using a delivery agent. In an aspect of the invention, the metadata may be RDF metadata. In another aspect of the invention, the match agent may determine the match using an RQL query.
Claim: The invention claimed is:

1. 1. A method of managing information in an information resource system having a server, a client, and a database, comprising:

- generating metadata using at least one import agent;
- determining at least one match using a match agent, wherein the at least one match is associated with at least one result; and
- dispatching the at least one result using a delivery agent.

Publication No. US 6643652 B2 (Update 200374 E)
Publication Date: 20031104
Method and apparatus for managing data exchange among systems in a network
Assignee: Saba Software, Inc., Redwood Shores, CA, US (SABA-N)
Inventor: Helgeson, Christopher S., Mountain View, CA, US
Lipkin, Daniel S., Belmont, CA, US
Larson, Robert S., Redwood City, CA, US
Panuganti, Srinivas, Sunnyvale, CA, US
Agent: Morrison & Foerster LLP, US
Language: EN
Application: US 2000176137 P 20000114 (Related to Provisional)
US 2001759678 A 20010112 (Local application)
Original US Class (secondary): 70710 707104.1 709202 709203
Original Abstract: The present mechanism provides a solution to the needs described above through a system and method for managing data exchange among systems in a network. The systems and methods of the present mechanism translate data from a system specific local format to a generic interchange format object, and vice versa, with predefined stylesheets using generic components and a system specific service components which utilize a native application programming interface of the specific local system.
Claim: What is claimed is:

1. 1. A method for managing data exchange between systems connected via a network, comprising:
 - creating a plurality of predefined stylesheets, each said stylesheet of said plurality of stylesheets describing a mapping between a system specific local format and a generic interchange format; wherein the generic interchange format is independent of a specific platform or system;
 - receiving a data object from a first system in a first system specific local format; translating said data object from a first system specific local format to a generic interchange format object with a first predefined stylesheet using a system specific service component which utilizes a native application programming interface of said first system;
 - translating said data object from said generic interchange format to a second system specific local format object with a second predefined stylesheet using a system specific service component which utilizes a native application programming interface of said second system; and
 - transferring said second system specific local format object to said second system.

Publication No. US 6721747 B2 (Update 200425 E)
Publication Date: 20040413
Method and apparatus for an information server
Assignee: Saba Software, Inc., Redwood Shores, CA, US
Inventor: Lipkin, Daniel S., Belmont, CA, US
Agent: Morrison & Foerster LLP, US
Language: EN

Application: US 2000176137 P 20000114 (Related to Provisional)
US 2001760432 A 20010112 (Local application)
Original US Class (secondary): 70710 7073 7078 707100 707200 709200 709202 709217 709225 715501.1 715513 715523
Original Abstract: The present invention provides a method and apparatus for managing information in an information resource system containing a server, a client, and a database, by generating metadata using an import agent, determining at least one match using a match agent, and dispatching the at least one match or a result associated with the match using a delivery agent. In an aspect of the invention, the metadata may be RDF metadata. In another aspect of the invention, the match agent may determine the match using an RQL query.
Claim: The invention claimed is:

1. 1. A method of managing information in an information resource system having a server, a client, and a database, comprising the acts of:
 - generating metadata using at least one import agent, wherein the import agent generates the metadata from one or more of a group consisting of existing web pages, business objects, content management systems and enterprise applications, and puts the generated metadata into a metadata database;
 - receiving a request to query the metadata database;
 - using a match agent to query the metadata and to pass a result of the query to a delivery agent;, and
 - dispatching the result of the query using the delivery agent, wherein the delivery agent interfaces with one or more of a group of delivery mechanisms comprising web page generators, XML datagrams, email and event messaging systems.

WIPO

Publication No. WO 2001052118 A2 (Update 200157 B)
Publication Date: 20010719
METHOD AND APPARATUS FOR AN INFORMATION SERVER
PROCEDE ET APPAREIL POUR SERVEUR D'INFORMATION
Assignee: *(except US)* SABA SOFTWARE, INC., 2400 Bridge Parkway, Redwood Shores, CA 94065, US Residence: US Nationality: US (SABA-N)
(only US) LIPKIN, Daniel, S., 309 Malcolm Avenue, Belmont, CA 94002, US Residence: US Nationality: US
Inventor: LIPKIN, Daniel, S., 309 Malcolm Avenue, Belmont, CA 94002, US Residence: US Nationality: US
Agent: HSIEH, Peter, Morrison & Foerster LLP, 425 Market Street, San Francisco, CA 94105-2482, US
Language: EN (233 pages, 17 drawings)
Application: WO 2001US920 A 20010112 (Local application)
Priority: US 2000176137 P 20000114
Designated States: (National Original) AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CR CU CZ DE DK DM DZ EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM TR TT TZ UA UG US UZ VN YU ZA ZW
(Regional Original) AT BE CH CY DE DK EA ES FI FR GB GH GM GR IE IT KE LS LU MC MW MZ NL OA PT SD SE SL SZ TR TZ UG ZW
Original IPC: G06F-17/30(A)
Current IPC: G06F-17/30(A)
Original Abstract: The present invention provides a method and apparatus for managing information in an

information resource system containing a server, a client, and a database, by generating metadata using an import agent, determining at least one match using a match agent, and dispatching the at least one match or a result associated with the match using a delivery agent. In an aspect of the invention, the metadata may be RDF metadata. In another aspect of the invention, the match agent may determine the match using an RQL query.

US006643652B2

(12) **United States Patent**
Helgeson et al.

(10) **Patent No.: US 6,643,652 B2**
(45) **Date of Patent: Nov. 4, 2003**

(54) **METHOD AND APPARATUS FOR MANAGING DATA EXCHANGE AMONG SYSTEMS IN A NETWORK**

(75) Inventors: **Christopher S. Helgeson**, Mountain View, CA (US); **Daniel S. Lipkin**, Belmont, CA (US); **Robert S. Larson**, Redwood City, CA (US); **Srinivas Panuganti**, Sunnyvale, CA (US)

(73) Assignee: **Saba Software, Inc.**, Redwood Shores, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 73 days.

(21) Appl. No.: **09/759,678**

(22) Filed: **Jan. 12, 2001**

(65) **Prior Publication Data**

US 2002/0073236 A1 Jun. 13, 2002

Related U.S. Application Data

(60) Provisional application No. 60/176,137, filed on Jan. 14, 2000.

(51) **Int. Cl.**[7] **G06F 17/30**

(52) **U.S. Cl.** **707/10**; 707/104.1; 709/202; 709/203

(58) **Field of Search** 707/1, 2, 3, 4, 707/8.9, 10, 102, 100, 103 R; 709/203, 202; 717/137, 143; 713/182; 705/8, 9, 76; 434/118, 362, 350, 217, 107; 700/7

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,701,130	A	10/1987	Whitney et al.	434/118
4,895,518	A	1/1990	Arnold et al.	434/118
5,119,465	A	6/1992	Jack et al.	717/137
5,199,068	A	3/1993	Cox	713/182
5,259,766	A	11/1993	Sack et al.	434/362
5,267,865	A	12/1993	Lee et al.	434/350
5,270,920	A	12/1993	Pearse et al.	705/8
5,301,270	A	4/1994	Steinberg et al.	345/866
5,310,349	A	5/1994	Daniels et al.	434/350
5,311,422	A	5/1994	Loftin et al.	703/2
5,326,270	A	7/1994	Ostby et al.	434/362
5,381,332	A	1/1995	Wood	705/8
5,395,243	A	3/1995	Lubin et al.	703/2
5,421,730	A	6/1995	Lasker, III et al.	434/118
5,499,293	A	3/1996	Behram et al.	705/76
5,548,506	A	8/1996	Srinivasan	705/8
5,597,312	A	1/1997	Bloom et al.	434/362
5,655,118	A	8/1997	Heindel et al.	707/103 R
5,761,063	A	6/1998	Jannette et al.	700/7
5,765,140	A	6/1998	Knudson et al.	705/9
5,788,504	A	8/1998	Rice et al.	434/217
5,795,155	A	8/1998	Morrel-Samuels	434/107
5,826,252	A	10/1998	Wolters, Jr. et al.	707/1
5,829,983	A	11/1998	Koyama et al.	434/118
5,832,483	A	11/1998	Barker	707/8
5,974,443	A	10/1999	Jeske	709/202
6,012,098	A	1/2000	Bayeh et al.	709/246
6,044,373	A	3/2000	Gladney et al.	707/10
6,125,363	A	9/2000	Buzzeo et al.	707/100
6,154,738	A *	11/2000	Call	707/4
6,446,256	B1 *	9/2002	Hyman et al.	717/143
6,463,440	B1 *	10/2002	Hind et al.	707/102

FOREIGN PATENT DOCUMENTS

WO	97/22096	6/1997
WO	98/05018	2/1998
WO	98/10399	3/1998
WO	98/24020	6/1998
WO	WO 99/27460	6/1999

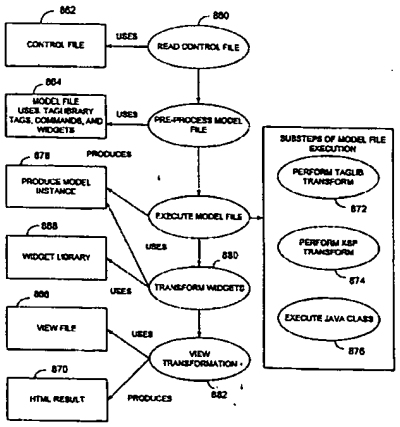
* cited by examiner

Primary Examiner—Diane D. Mizrahi
Assistant Examiner—Yicun Wu
(74) *Attorney, Agent, or Firm*—Morrison & Foerster LLP

(57) **ABSTRACT**

The present mechanism provides a solution to the needs described above through a system and method for managing data exchange among systems in a network. The systems and methods of the present mechanism translate data from a system specific local format to a generic interchange format object, and vice versa, with predefined stylesheets using generic components and a system specific service components which utilize a native application programming interface of the specific local system.

31 Claims, 17 Drawing Sheets



What is claimed is:

1. A method for managing data exchange between systems connected via a network, comprising:

creating a plurality of predefined stylesheets, each said stylesheet of said plurality of stylesheets describing a mapping between a system specific local format and a generic interchange format; wherein the generic interchange format is independent of a specific platform or system;

receiving a data object from a first system in a first system specific local format; translating said data object from a first system specific local format to a generic interchange format object with a first predefined stylesheet using a system specific service component which utilizes a native application programming interface of said first system;

translating said data object from said generic interchange format to a second system specific local format object with a second predefined stylesheet using a system specific service component which utilizes a native application programming interface of said second system; and

transferring said second system specific local format object to said second system.

2. The method of claim 1, wherein said step of receiving a data object from a first system in a first system specific local format comprises:

receiving a request to export a data object from a first system;

identifying a local data object identifier utilizing a mapper component;

identifying a document type utilizing a mapper component;

identifying a stylesheet and transformer mechanism using said document type; and

extracting said data object from said first system.

3. The method of claim 2, further comprising converting said local data object identifier to a platform object identifier.

4. The method of claim 1, wherein said step of translating said data object from said generic interchange format to a second system specific local format object with said second predefined stylesheet using a system specific service component which utilizes a native application programming interface of said second system comprises:

receiving a request to import a data object to a second system;

receiving a data object in a generic interchange format, a document type, and a platform object identifier;

identifying a stylesheet and transformer mechanism using said document type; and

translating said data object from said generic interchange format to a second system specific local format object with said stylesheet using a system specific service component which utilizes a native application programming interface of said second system.

5. The method of claim 4, further comprising:

scanning for foreign platform object identifiers; and

resolving said foreign platform object identifiers to a local identifier.

6. The method of claim 1, further comprising returning a local identifier for said data object transferred to said second system.

7. The method of claim 1, further comprising the step:

monitoring a first system for changes to a data object at said first system

8. The method of claim 1, wherein said step of receiving a data object from a first system in a first system specific local format comprises extracting said data object from said first system with a system specific service component which utilizes a native programming interface of said first system.

9. The method of claim 1, wherein said step of transferring said second system specific local format object may be performed using a plurality of communication protocols.

10. The method of claim 1, wherein said first predefined stylesheet is an xsl stylesheet.

11. The method of claim 1, wherein said data objects are in xml format.

12. The method of claim 1, wherein said step of translating said data object from a first system specific local format to a generic interchange format object with said first predefined stylesheet using a system specific service component which utilizes a native application programming interface of said first system comprises:

translating said data object into a serialized local XML representation; and

translating said serialized local XML representation to a generic interchange format utilizing a predefined stylesheet.

13. The method of claim 1, wherein said step of translating said data object from said generic interchange format to a second system specific local format object with said second predefined stylesheet using a system specific service component which utilizes a native application programming interface of said second system comprises:

mapping said data object in generic interchange format to one or more objects required to be transferred to said second system; and

translating said generic interchange format data into said second system specific local format using a predefined stylesheet.

14. An apparatus for managing data exchange between systems connected via a network, comprising:

a network interface;

memory storing data and programs of instructions;

a processor coupled to the memory which executes the programs of instructions and accesses the stored data, wherein the programs of instructions comprise:

a first translator component for translating a data object from a first system specific local format to a generic interchange format object, wherein said generic interchange format object is both platform and system independent, said first translator component comprising:

a system independent service subcomponent; and

a system specific service component utilizing a native API of said second system to translate said data object from a generic interchange format object to a second system specific local format object using a second predefined stylesheet; and

a second translator component for translating said data object from said generic interchange format to a second system specific local format object, said second translator component comprising:

a system independent service subcomponent; and

a system specific service component utilizing a native API of said second system to translate said data object from a generic interchange format object to a second system specific local format object using a second predefined stylesheet; and

a delivery component for transferring said data object between said first and second system.

15. The apparatus of claim 14, wherein said programs of instructions further comprise:

a monitor component for monitoring changes of a data object at a first system, said monitor component comprising:

a system independent service subcomponent; and

a system specific service component utilizing a native API of said first system to monitor changes of said data object at said first system.

16. The apparatus of claim 14, wherein said programs of instructions further comprise a mapper component for identifying a local object identifier and a document type.

17. The apparatus of claim 14, wherein said first predefined stylesheet is an xsl stylesheet.

18. The apparatus of claim 14, wherein said data objects are in xml format.

19. A method for managing data exchange between systems connected via a network, comprising:

creating a plurality of predefined stylesheets, each said stylesheet of said plurality of stylesheets describing a mapping between a system specific local format and a generic interchange format;

receiving a data object from a first system in a first system specific local format;

translating said data object from a first system specific local format to a generic interchange format object with said predefined stylesheets using a system specific service component which utilizes a native application programming interface of said first system;

translating said data object from said generic interchange format to a second system specific local format object with said predefined stylesheets using a system specific service component which utilizes a native application programming interface of said second system comprising;

receiving a request to import a data object to a second system;

receiving a data object in a generic interchange format, a document type, and a platform object identifier;

scanning for foreign platform object identifiers;

resolving said foreign platform object identifiers to a local identifier;

identifying a stylesheet and transformer using said document type; and

translating said data object from said generic interchange format to a second system specific local format object with said stylesheet using a system specific service component which utilizes a native application programming interface of said second system; and

transferring said second system specific local format object to said second system.

20. The method of claim 19, further comprising returning a local identifier for said data object transferred to said second system.

21. The method of claim 19, further comprising the step:

monitoring a first system for changes to a data object at said first system.

22. The method of claim 19, wherein said step of receiving a data object from a first system in a first system specific local format comprises extracting said data object from said first system with a system specific service component which utilizes a native programming interface of said first system.

23. The method of claim 19, wherein said step of transferring said translated data object may be performed using a plurality of communication protocols.

24. The method of claim 19, wherein said stylesheet is an xsl stylesheet.

25. The method of claim 19, wherein said data objects are in xml format.

26. The method of claim 19, wherein said step of translating said data object from a first system specific local format to a generic interchange format object with said predefined stylesheets using a system specific service component which utilizes a native application programming interface of said first system comprises:

translating said data object into a serialized local XML representation; and

translating said serialized local XML representation to a generic interchange format utilizing a predefined stylesheet.

27. The method of claim 19, wherein said step of translating said data object from said generic interchange format to a second system specific local format object with said predefined stylesheets using a system specific service component which utilizes a native application programming interface of said second system comprises:

mapping said data object in generic interchange format to one or more objects required to be transferred to said second system; and

translating said generic interchange format data into said second system specific local format using a predefined stylesheet.

28. An apparatus for managing data exchange between systems connected via a network, comprising:

a network interface;

memory storing data and programs of instructions;

a processor coupled to the memory which executes the programs of instructions and accesses the stored data,

wherein the programs of instructions comprise:

a monitor component for monitoring changes of a data object at a first system,

said monitor component comprising:

a system independent service subcomponent; and

a system specific service component utilizing a native API of said first system to monitor changes of said data object at said first system;

a first translator component for translating a data object from a first system specific local format to a generic interchange format object, said first translator component comprising:

a system independent service subcomponent; and

a system specific service component utilizing a native API of said first system to translate said data object to a generic interchange format object using a predefined stylesheet;

a second translator component for translating said data object from said generic interchange format to a second system specific local format object, said second translator component comprising:

a system independent service subcomponent; and

a system specific service component utilizing a native API of said second system to translate said data object from a generic interchange format object to a second system specific local format object using a predefined stylesheet; and

a delivery component for transferring said data object between said first and second system.

29. The apparatus of claim 28, wherein said programs of instructions further comprise a mapper component for identifying a local object identifier and a document type.

30. The apparatus of claim 28, wherein said predefined stylesheet is an xsl stylesheet.

31. The apparatus of claim 28, wherein said data objects are in xml format.

* * * * *

65/7/9 (Item 9 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

0014448883 *Drawing available*
WPI Acc no: 2004-639770/200462
XRPX Acc No: N2004-505574
Voice mail message retrieval method in web based messaging system, involves transmitting hypertext markup language page having XML tags that specify playback of messages retrieved from subscriber's mailbox address by proxy browser
Patent Assignee: CISCO TECHNOLOGY INC (CISC-N)
Inventor: DODRILL L D; JOSHI S; WOLFE W M

Patent Family (1 patents, 1 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
US 6779025	B1	20040817	US 2000608187	A	20000630	200462	B

Priority Applications (no., kind, date): US 2000608187 A 20000630

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
US 6779025	B1	EN	13	4		

Alerting Abstract US B1
NOVELTY - An **application** server (66) acquires roaming subscriber's attribute information and **mailbox** address from **database** server, when proxy browser (62) initiates a messaging **application** with the server for the subscriber, after terminating call from the subscriber in the nearest switching center. The server generates HTML **page** containing **XML** tags that specify playback of messages read from the address, by the browser.
DESCRIPTION - An INDEPENDENT CLAIM is also included for computer readable recorded medium storing **program** for retrieving voice **mail** message for roaming wireless subscriber.
USE - In web based messaging system for retrieval of voice **mail** message for roaming wireless subscriber, for browser web **application**, electronic **mail application**, two-way instant messaging or voice over internet protocol (IP) phone calls using personal computer (PC), etc.
ADVANTAGE - The retrieval of the subscriber **attribute** information from remote server enables deployment of mobile internet for roaming PCs, global telephones, etc. Hence any device accessing a local server can obtain full **application** services based on the local server accessing the remote subscriber **attribute** information for that device, regardless of the **application** services. This also facilitates provision of further mobility for notification-based **application** services.
DESCRIPTION OF DRAWINGS - DESCRIPTION OF DRAWING - The figure shows a block diagram of the web based messaging system.
18a cordless telephone
18b facsimile
18c analog telephone

18d cellular telephone
62 proxy browser
64 web server
66 **application** server

Title Terms /Index Terms/Additional Words: VOICE; **MAIL**; MESSAGE; RETRIEVAL; METHOD; WEB; BASED; MESSAGING; SYSTEM; TRANSMIT; LANGUAGE; PAGE; TAG; SPECIFIED; PLAYBACK; SUBSCRIBER; **MAILBOX**; ADDRESS

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-013/00			Main		"Version 7"

US Classification, Issued: 709219000, 709206000, 709225000, 709249000

File Segment: EPI;
DWPI Class: T01; W01
Manual Codes (EPI/S-X): **T01-J18**; **T01-N03B2**; **T01-S03**; W01-C02B7C; W01-C05B4C; W01-C05B5E

Original Publication Data by Authority

United States

Publication No. US 6779025 B1 (Update 200462 B)
Publication Date: 20040817
IP web based messaging system for localized management of wireless messaging
Assignee: Cisco Technology, Inc., San Jose, CA, US (CISC-N)
Inventor: Wolfe, William M., Santa Monica, CA, US
Dodrill, Lewis Dean, Richmond, VA, US
Joshi, Satish, Glen Allen, VA, US
Agent: Turkevich, Leon R., US
Language: EN (13 pages, 4 drawings)
Application: US 2000608187 A 20000630 (Local application)
Original IPC: G06F-13/00(A)
Current IPC: G06F-13/00(A)
Original US Class (main): 709219
Original US Class (secondary): 709206 709225 709249
Original Abstract: An application server, configured for executing a messaging application defined by XML documents, is able to provide messaging services to a calling party for a remote subscriber based on retrieval of subscriber attribute information obtained from a remote information database server (such as LDAP) via an Internet Protocol (IP) network. The application server, upon receiving a voice application operation request having subscriber identifier information from a browser serving the calling party, obtains from the remote information

database server attribute information for responding to the voice application operation request based on the subscriber identifier information. The application server generates an HTML document having media content and control tags for personalized execution of the voice application operation based on the attribute information obtained from the standardized database server. The application server also is configured for sending and retrieving messages from a standardized remote messaging server via the IP network.
Claim: What is claimed is:

1. 1. A method in a wireless communications system of retrieving a stored message for a roaming wireless subscriber, the method including:
 - terminating by a mobile switching center an incoming wireless telephone call from the roaming wireless subscriber to a proxy device configured for generating HTTP requests and playback of media content based on selective execution of tagged information within web pages;
 - initiating by the proxy device a messaging application session executed by an application server for the roaming wireless subscriber, the application server configured for generating the web pages based on the respective HTTP requests, at least one of the HTTP requests specifying the wireless subscriber;
 - obtaining by the application server subscriber attribute information related to the roaming wireless subscriber from an Internet Protocol (IP) based database server via an IP network;
 - obtaining, by the application server, the stored message via the IP network from an IP-based messaging server based on a prescribed address retrieved from the obtained subscriber attribute information; and
 - generating by the application server a web page including tags for playback of the stored message to the roaming wireless subscriber by the proxy device.

US006779025B1

(12) **United States Patent**
Wolfe et al.

(10) **Patent No.: US 6,779,025 B1**
(45) **Date of Patent: Aug. 17, 2004**

(54) **IP WEB BASED MESSAGING SYSTEM FOR LOCALIZED MANAGEMENT OF WIRELESS MESSAGING**

(75) Inventors: **William M. Wolfe**, Santa Monica, CA (US); **Lewis Dean Dodrill**, Richmond, VA (US); **Satish Joshi**, Glen Allen, VA (US)

(73) Assignee: **Cisco Technology, Inc.**, San Jose, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 514 days.

(21) Appl. No.: **09/608,187**

(22) Filed: **Jun. 30, 2000**

(51) **Int. Cl.**[7] **G06F 13/00**
(52) **U.S. Cl.** **709/219**; 709/206; 709/225; 709/249
(58) **Field of Search** 709/203, 206, 709/217, 219, 223, 225, 229, 249; 719/311, 313, 328

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,837,798	A	6/1989	Cohen et al.	
6,061,432	A	5/2000	Wallace et al.	
6,138,158	A *	10/2000	Boyle et al.	709/225
6,311,194	B1 *	10/2001	Sheth et al.	707/505
6,385,202	B1 *	5/2002	Katseff et al.	370/401
6,389,276	B1 *	5/2002	Brilla et al.	455/413
6,400,940	B1 *	6/2002	Sennett	455/414
6,519,246	B1 *	2/2003	Strahs	370/259
6,556,997	B1 *	4/2003	Levy	707/10

OTHER PUBLICATIONS

Sims, "Standards Update: The Voice Browser", Oreilly Network, Mar. 11, 2000, 4 pages.
Cox, "Allowing The Web To Be Heard", Network World Fusion, Apr. 24, 2000, 2 pages.

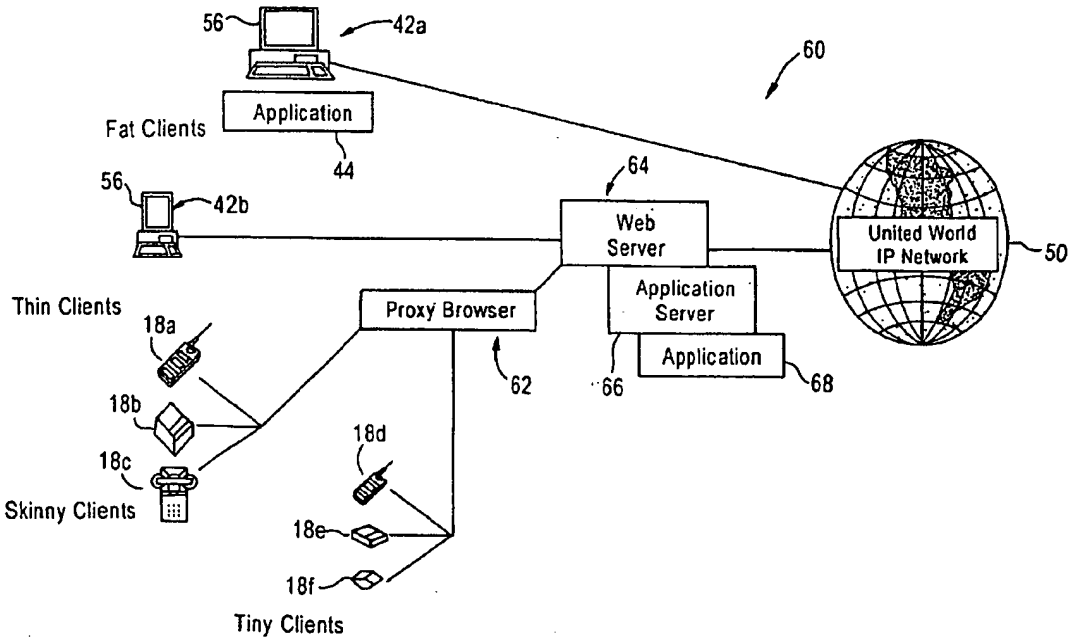
* cited by examiner

Primary Examiner—Viet D. Vu
(74) *Attorney, Agent, or Firm*—Leon R. Turkevich

(57) **ABSTRACT**

An application server, configured for executing a messaging application defined by XML documents, is able to provide messaging services to a calling party for a remote subscriber based on retrieval of subscriber attribute information obtained from a remote information database server (such as LDAP) via an Internet Protocol (IP) network. The application server, upon receiving a voice application operation request having subscriber identifier information from a browser serving the calling party, obtains from the remote information database server attribute information for responding to the voice application operation request based on the subscriber identifier information. The application server generates an HTML document having media content and control tags for personalized execution of the voice application operation based on the attribute information obtained from the standardized database server. The application server also is configured for sending and retrieving messages from a standardized remote messaging server via the IP network .

12 Claims, 5 Drawing Sheets



ported to different application servers based on remote access of subscriber profile information and subscriber message stores. Hence, unified messaging systems can utilize the IP network for transport of messages, and sharing of application infrastructure with PCs, browser web applications, and e-mail applications. Moreover, the retrieval of subscriber profile information from remote profile directories enables deployment of a mobile Internet for roaming PCs, global telephones, e-mail applications, etc. Hence, any device accessing a local server can obtain full application services based on the local server accessing the remote subscriber profile information for that device, regardless of whether the application services are for messaging applications, two-way instant messaging or real-time voice over IP phone calls using a PC, etc. In addition, the retrieval of subscriber profile information enables notification systems to follow the subscriber, providing further mobility for notification-based application services.

While this invention has been described in connection with what is presently considered to be the most practical and preferred embodiment, it is to be understood that the invention is not limited to the disclosed embodiments, but, on the contrary, is intended to cover various modifications and equivalent arrangements included within the spirit and scope of the appended claims.

What is claimed is:

1. A method in a wireless communications system of retrieving a stored message for a roaming wireless subscriber, the method including:

terminating by a mobile switching center an incoming wireless telephone call from the roaming wireless subscriber to a proxy device configured for generating HTTP requests and playback of media content based on selective execution of tagged information within web pages;

initiating by the proxy device a messaging application session executed by an application server for the roaming wireless subscriber, the application server configured for generating the web pages based on the respective HTTP requests, at least one of the HTTP requests specifying the wireless subscriber;

obtaining by the application server subscriber attribute information related to the roaming wireless subscriber from an Internet Protocol (IP) based database server via an IP network;

obtaining, by the application server, the stored message via the IP network from an IP-based messaging server based on a prescribed address retrieved from the obtained subscriber attribute information; and

generating by the application server a web page including tags for playback of the stored message to the roaming wireless subscriber by the proxy device.

2. The method of claim 1, wherein the generating step includes adding media tags within the web page that specify prescribed audible prompts based on language preferences specified in the subscriber attribute information.

3. The method of claim 1, with further comprising storing the subscriber attribute information in a local registry for use by the application server during the messaging application session with the roaming wireless subscriber.

4. The method of claim 1, wherein the generating step includes outputting the web page to the proxy device based on validating the roaming wireless subscriber relative to a password supplied by the proxy device and a stored password within the subscriber attribute information.

5. The method of claim 1, wherein the step of obtaining the subscriber attribute information includes accessing the IP based database server according to LDAP protocol.

6. The method of claim 1, wherein the step of obtaining the stored message includes accessing the IP-based messaging server according to IMAP protocol.

7. A computer readable medium having stored thereon sequences of instructions for retrieving a stored message for a roaming wireless subscriber, the sequences of instructions including instructions for performing the steps of:

terminating by a mobile switching center an incoming wireless telephone call from the roaming wireless subscriber to a proxy device configured for generating HTTP requests and playback of media content based on selective execution of tagged information within web pages;

initiating by the proxy device a messaging application session executed by an application server for the roaming wireless subscriber, the application server configured for generating the web pages based on the respective HTTP requests, at least one of the HTTP requests specifying the wireless subscriber;

obtaining by the application server subscriber attribute information related to the roaming wireless subscriber from an Internet Protocol (IP) based database server via an IP network;

obtaining, by the application server, the stored message via the IP network from an IP-based messaging server based on a prescribed address retrieved from the obtained subscriber attribute information; and

generating by the application server a web page including tags for playback of the stored message to the roaming wireless subscriber by the proxy device.

8. The medium of claim 7, wherein the generating step includes adding media tags within the Web page that specify prescribed audible prompts based on language preferences specified in the subscriber attribute information.

9. The medium of claim 7, with further comprising instructions for performing the step of storing the subscriber attribute information in a local registry for use by the application server during the messaging application session with the roaming wireless subscriber.

10. The medium of claim 7, wherein the generating step includes outputting the web page to the proxy device based on validating the roaming wireless subscriber relative to a password supplied by the proxy device and a stored password within the subscriber attribute information.

11. The medium of claim 7, wherein the step of obtaining the subscriber attribute information includes accessing the IP based database server according to LDAP protocol.

12. The medium of claim 7, wherein the step of obtaining the stored message includes accessing the IP-based messaging server according to IMAP protocol.

* * * * *

65/7/2 (Item 2 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

0015646630 *Drawing available*
WPI Acc no: 2006-210809/200622
Related WPI Acc No: 2003-147216; 2003-265606; 2003-656232; 2004-313511; 2004-485761; 2004-591078; 2004-781947; 2005-119795; 2005-401610; 2005-494777; 2005-617185; 2005-617492; 2005-699537; 2005-710219; 2005-766217; 2006-115442
XRPX Acc No: N2006-181342

Method of executing voice messaging application in application server, involves generating hypertext transfer protocol page having markup language tags that specify media content and control information for playing media files

Patent Assignee: BARBAN S H (BARB-I); DANNER R A (DANN-I); DODRILL L D (DODR-I); JOSHI S (JOSH-I); MARTIN S J (MART-I); RAVISHANKAR G (RAVI-I); RAVISHANKAR S (RAVI-I)
Inventor: BARBAN S H; DANNER R A; DODRILL L D; JOSHI S; MARTIN S J; RAVISHANKAR G; RAVISHANKAR S

Patent Family (1 patents, 1 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
US 20060047767	A1	20060302	US 1999152316	P	19990903	200622	B
			US 2000588293	A	20000607		
			US 2005219844	A	20050907		

CISCO

Priority Applications (no., kind, date): US 2000588293 A 20000607; US 1999152316 P 19990903; US 2005219844 A 20050907

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
US 20060047767	A1	EN	17	4	Related to Provisional	US 1999152316
					Continuation of application	US 2000588293

Alerting Abstract US A1

NOVELTY - A selected **extensible markup language (XML) document** is accessed based on parameters specified within hypertext **transfer** protocol (HTTP) **request** received from web browser, and based on **application** state determined from accessing a brownie. The **XML** tags within accessed **XML documents** are parsed and HTML **page** having **XML** tags that specify media content and control information for playing files by corresponding browser is generated.

DESCRIPTION - INDEPENDENT CLAIMS are also included for the following:

1. **application** server; and
2. **computer** readable medium storing instructions for executing voice messaging **application**.

USE - For executing voice messaging **application** in **application** server (claimed) in an internet protocol (IP) network.
ADVANTAGE - Enables users to customize unified messaging systems without the necessity of public switched telephone network based voice **mail** systems.
DESCRIPTION OF DRAWINGS - The figure shows a block diagram of the **application** server.
66 **application** server

Title Terms /Index Terms/Additional Words: METHOD; EXECUTE; VOICE; MESSAGING; APPLY; SERVE; GENERATE; **TRANSFER**; PROTOCOL; PAGE; LANGUAGE; TAG; SPECIFIED; MEDIUM; CONTENT; CONTROL; INFORMATION; PLAY; **FILE**

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-0015/16	A	I	F	B	20060101

US Classification, Issued: 709206000

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): **T01-J11A1**; **T01-N01D1A**; **T01-N03A1**; **T01-S03**

Original Publication Data by Authority

United States

Publication No. US 20060047767 A1 (Update 200622 B)
Publication Date: 20060302
Unified messaging system using web based application server for management of messages using standardized servers
Assignee: Dodrill, Lewis Dean, Richmond, VA, US Residence: US Nationality: US (DODR-I)
Ravishankar, Geetha, Glen Allen, VA, US Residence: US Nationality: US (RAVI-I)
Joshi, Satish, Glen Allen, VA, US Residence: US Nationality: US (JOSH-I)
Danner, Ryan Alan, Glen Allen, VA, US Residence: US Nationality: US (DANN-I)
Barban, Susan Harrow, Midlothian, VA, US Residence: US Nationality: US (BARB-I)
Martin, Steven J., Richmond, VA, US Residence: US Nationality: US (MART-I)
Ravishankar, Swaminathan, Glen Allen, VA, US Residence: US Nationality: US (RAVI-I)
Inventor: Dodrill, Lewis Dean, Richmond, VA, US Residence: US Nationality: US
Ravishankar, Geetha, Glen Allen, VA, US Residence: US Nationality: US
Joshi, Satish, Glen Allen, VA, US Residence: US Nationality: US
Danner, Ryan Alan, Glen Allen, VA, US Residence: US Nationality: US
Barban, Susan Harrow, Midlothian, VA, US Residence: US Nationality: US

Martin, Steven J., Richmond, VA, US Residence: US Nationality: US
Ravishankar, Swaminathan, Glen Allen, VA, US Residence: US Nationality: US
Agent: LEON R TURKEVICH, 2000 M STREET NW, 7TH FLOOR, WASHINGTON, DC, US
Language: EN (17 pages, 4 drawings)
Application: US 1999152316 P 19990903 (Related to Provisional)
US 2000588293 A 20000607 (Continuation of application)
US 2005219844 A 20050907 (Local application)
Original IPC: G06F-15/16(B,I,H,US,20060101,20060302,A,F)
Current IPC: G06F-15/16(B,I,H,US,20060101,20060302,A,F)
Original US Class (secondary): 709206
Original Abstract: A unified web-based voice messaging system uses an application server, configured for executing a voice application defined by XML documents, that accesses subscriber attributes from a standardized information database server (such as LDAP), and messages from a standardized messaging server (such as IMAP), regardless of message format. The application server, upon receiving a request from a browser serving a user, accesses the standardized database server to obtain attribute information for responding to the voice application operation request. The application server generates an HTML document having media content and control tags for personalized execution of the voice application operation based on the attribute information obtained from the standardized database server. The application server also is configured for storing messages for a called party in the standardized messaging server by storing within the message format information that specifies the corresponding message format. Hence, the application server can respond to a request for a stored message from a subscriber by accessing the stored message from the standardized messaging server, and generating an HTML document having media content and control tags for presenting the subscriber with the stored message in a prescribed format based on the message format and the capabilities of the access device used by the subscriber.
Claim:

1. 1. A method in an application server for executing a voice messaging application, the method comprising:
 - receiving a first HTTP request for execution of a prescribed voice messaging application operation for a subscriber;
 - accessing attribute information for the subscriber from an Internet Protocol (IP) based database server configured for storing subscriber attributes;
 - accessing an IP-based messaging server for subscriber messaging information based on the accessed attribute information, each stored message on the IP-based messaging server being stored within a corresponding e-mail message as a URL encoded string with the corresponding header information so that each stored message is encoded in the URL encoded string; and
 - generating an HTML page, for execution of the prescribed voice messaging application operation and having media content and control tags, based on the first HTTP request and the subscriber messaging information.

US 20060047767A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2006/0047767 A1

Dodrill et al. (43) Pub. Date: Mar. 2, 2006

(54) UNIFIED MESSAGING SYSTEM USING WEB BASED APPLICATION SERVER FOR MANAGEMENT OF MESSAGES USING STANDARDIZED SERVERS

(76) Inventors: **Lewis Dean Dodrill**, Richmond, VA (US); **Geetha Ravishankar**, Glen Allen, VA (US); **Satish Joshi**, Glen Allen, VA (US); **Ryan Alan Danner**, Glen Allen, VA (US); **Susan Harrow Barban**, Midlothian, VA (US); **Steven J. Martin**, Richmond, VA (US); **Swaminathan Ravishankar**, Glen Allen, VA (US)

Correspondence Address:
LEON R TURKEVICH
2000 M STREET NW
7TH FLOOR
WASHINGTON, DC 200363307

CISCO

(21) Appl. No.: 11/219,844

(22) Filed: Sep. 7, 2005

Related U.S. Application Data

(63) Continuation of application No. 09/588,293, filed on Jun. 7, 2000.

(60) Provisional application No. 60/152,316, filed on Sep. 3, 1999.

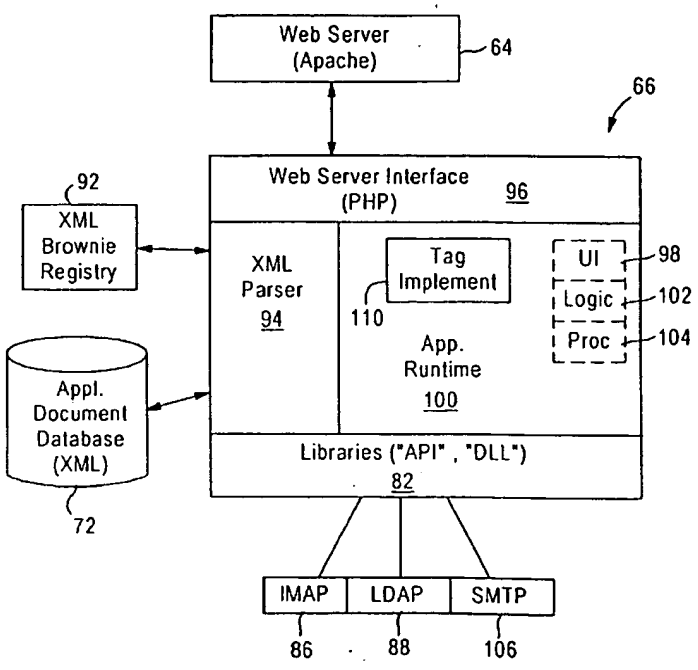
Publication Classification

(51) Int. Cl.
G06F 15/16 (2006.01)

(52) U.S. Cl. 709/206

(57) ABSTRACT

A unified web-based voice messaging system uses an application server, configured for executing a voice application defined by XML documents, that accesses subscriber attributes from a standardized information database server (such as LDAP), and messages from a standardized messaging server (such as IMAP), regardless of message format. The application server, upon receiving a request from a browser serving a user, accesses the standardized database server to obtain attribute information for responding to the voice application operation request. The application server generates an HTML document having media content and control tags for personalized execution of the voice application operation based on the attribute information obtained from the standardized database server. The application server also is configured for storing messages for a called party in the standardized messaging server by storing within the message format information that specifies the corresponding message format. Hence, the application server can respond to a request for a stored message from a subscriber by accessing the stored message from the standardized messaging server, and generating an HTML document having media content and control tags for presenting the subscriber with the stored message in a prescribed format based on the message format and the capabilities of the access device used by the subscriber.



1. A method in an application server for executing a voice messaging application, the method comprising:

receiving a first HTTP request for execution of a prescribed voice messaging application operation for a subscriber;

accessing attribute information for the subscriber from an Internet Protocol (IP) based database server configured for storing subscriber attributes;

accessing an IP-based messaging server for subscriber messaging information based on the accessed attribute information, each stored message on the IP-based messaging server being stored within a corresponding e-mail message as a URL encoded string with the corresponding header information so that each stored message is encoded in the URL encoded string; and

generating an HTML page, for execution of the prescribed voice messaging application operation and having media content and control tags, based on the first HTTP request and the subscriber messaging information.

2. The method of claim 1, wherein the receiving step includes recovering within the HTTP request a browser configuration, and call parameters.

3. The method of claim 2, wherein the recovering step includes identifying the browser configuration as one of a computer browser configuration configured for parsing a prescribed group of media tags and presenting a prescribed group of media types, and a lightweight browser configuration configured for parsing a prescribed portion of the prescribed group of media tags.

4. The method of claim 3, wherein the generating step includes generating the HTML page by selectively supplying media tag types based on the identified browser configuration. Page 3

5. The method of claim 2, wherein the call parameters include a called party identifier, the accessing step including retrieving the attribute information, specifying at least one of subscriber registration status and subscriber messaging preferences, based on the called party identifier.

6. The method of claim 5, wherein the call parameters include a calling party identifier, the accessing step further including retrieving second subscriber attribute information based on the calling party identifier.

7. (canceled)

8. The method of claim 1, wherein the accessing step includes accessing the IP-based database server according to LDAP protocol.

9. The method of claim 1, wherein the step of accessing the IP-based messaging server includes selectively obtaining from the IP-based messaging server at least one of a subscriber name, and a subscriber greeting as a subscriber prompt based on a subscriber identifier obtained from the accessed attribute information.

10. The method of claim 9, wherein the converting step includes converting the subscriber prompt from the corresponding URL encoded string into a media file having at least one prescribed media type.

11. The method of claim 10, wherein the converting step includes converting the subscriber prompt into a Multipurpose Internet Media Extension (MIME) type .wav file playable by a browser.

12. The method of claim 11, wherein the step of generating an HTML page includes inserting a first media tag including the .wav file and a second media tag configured for controlling playing of the .wav file.

13. The method of claim 1, wherein the step of accessing the IP-based messaging server includes determining a presence of a stored message on the IP-based messaging server for the subscriber based on the subscriber messaging information, the generating step including selectively inserting one of a first prompt file specifying no new messages and a second prompt file specifying the determined presence of the stored message, based on the subscriber messaging information.

14. The method of claim 13, wherein the step of accessing the IP-based messaging server further includes identifying, for each stored message, a corresponding message type based on the corresponding header information specifying the a Multipurpose Internet Media Extension (MIME) type, the second prompt file configured for specifying the corresponding message type for each stored message.

15. The method of claim 14, further comprising:

selecting one of the stored messages from the IP-based messaging server;

the converting step including converting the URL encoded string of the selected one message into a media file having a prescribed media type, based on the corresponding MIME type and determined capabilities of a browser having sent the first HTTP request, the generating step including inserting the media file into a media tag with a corresponding media control tag for playback of the media file by the browser.

16. The method of claim 15, wherein the converting step includes converting the URL encoded string to text and executing a text to speech routine for converting the text into an audio file based on the header information specifying text and the determined attributes specifying audio only.

17. (canceled)

18. The method of claim 14, wherein

the converting step includes converting selected header information into an audio file based on determining the MIME type is incompatible with determined capabilities of the browser, the generating step including inserting the audio file into the HTML page for playback by the browser.

19. (canceled)

20. The method of claim 1, further comprising:

receiving a second HTTP request for storage for the subscriber of a message having a prescribed messaging format; and

outputting to the IP-based messaging server an instruction for storage of a standard-format message, containing the message and header information specifying the prescribed messaging format, in a directory specified for the subscriber.

21. The method of claim 20, wherein the outputting step includes:

converting the message into the corresponding a URL encoded string;

generating a the corresponding header that specifies a Multipurpose Internet Media Extension (MIME) type for the prescribed messaging format; and

sending as the standard-format message an e-mail message, including the URL encoded string and the header as an attachment, to the IP-based messaging server according to SMTP protocol for delivery to the directory specified for the subscriber.

22. An application server configured for executing a voice messaging application, the application server including:

a hypertext transport protocol (HTTP) interface for receiving an HTTP request specifying execution of a prescribed voice messaging application operation for a subscriber; and

an application runtime environment configured for dynamically generating, in response to the HTTP request, a first hypertext markup language (HTML) document having media content for execution of the voice messaging application operation for the subscriber based on accessing attribute information for the subscriber from an Internet Protocol (IP) based database server configured for storing subscriber attributes, and based on accessing an IP-based messaging server for subscriber messaging information based on the accessed attribute information,

wherein each stored message on the IP-based messaging server is stored within a corresponding e-mail message as a URL encoded string with the corresponding header information so that each stored message is encoded in the URL encoded string.

23. The server of claim 22, wherein the application runtime environment is configured for determining subscriber registration status and subscriber messaging preferences in response to a called party identifier specified in the HTTP request.

24. The server of claim 23, wherein the application runtime environment accesses the IP-based database server and the IP-based messaging server according to LDAP protocol and IMAP protocol, respectively.

25. The server of claim 24, wherein the application runtime environment is configured for accessing from the IP-based messaging server at least one of a subscriber name and a subscriber greeting as a subscriber prompt based on the HTTP request specifying a condition for a calling party to leave a message for the subscriber, the application runtime environment converting the subscriber prompt into a media file playable by the a browser and inserting the media file into the HTML document.

26. The server of claim 25, wherein the application runtime environment is configured for converting the subscriber prompt stored on the IP-based messaging server from a URL encoded string into an audio file as said media file.

27. The server of claim 24, wherein the application runtime environment is configured for accessing from the IP-based messaging server a message for a subscriber, stored on the IP-based messaging server as an e-mail message having the URL encoded string and corresponding header information that specifies a Multipurpose Internet Media Extension (MIME) type, the application runtime environment converting at least one of the header information and the URL encoded string into a media file having a selected media type based on the MIME type and determined capabilities of an input device used by the subscriber.

28. The server of claim 27, wherein the application runtime environment is configured for converting the header information into an audio file based on determining that the MIME type specifies an image and that the determined capabilities to not support the image.

29. (canceled)

30. The server of claim 27, wherein the application runtime environment is configured for converting the URL encoded string into text and converting the text into an audio file using a text to speech routine, based on the MIME type specifying text and the determined capabilities do not support text.

31. The server of claim 24, wherein the application runtime environment is configured for converting a message supplied by the HTTP request and having a prescribed messaging format into the URL encoded string, generating a header specifying a Multipurpose Internet Media Extension (MIME) type for the prescribed messaging format, and outputting to the IP-based messaging server an e-mail message, including the URL encoded string and the header as an attachment, for delivery to a directory specified for the subscriber.

32. A computer readable medium having stored thereon sequences of instructions for executing a voice messaging application, the sequences of instructions including instructions for performing the steps of:

receiving a first HTTP request for execution of a prescribed voice messaging application operation for a subscriber;

accessing attribute information for the subscriber from an Internet Protocol (IP) based database server configured for storing subscriber attributes;

accessing an IP-based messaging server for subscriber messaging information based on the accessed attribute information, each stored message on the IP-based messaging server being stored within a corresponding e-mail message as a URL encoded string with the corresponding header information so that each stored message is encoded in the URL encoded string; and

generating an HTML page, for execution of the prescribed voice messaging application operation and having media content and control tags, based on the first HTTP request and the subscriber messaging information.

33. The medium of claim 32, wherein the receiving step includes recovering within the HTTP request a browser configuration, and call parameters.

34. The medium of claim 33, wherein the recovering step includes identifying the browser configuration as one of a computer browser configuration configured for parsing a prescribed group of media tags and presenting a prescribed group of media types, and a lightweight browser configuration configured for parsing a prescribed portion of the prescribed group of media tags.

35. The medium of claim 34, wherein the generating step includes generating the HTML page by selectively supplying media tag types based on the identified browser configuration.

36. The medium of claim 33, wherein the call parameters include a called party identifier, the accessing step including retrieving the attribute information, specifying at least one of subscriber registration status and subscriber messaging preferences, based on the called party identifier.

37. The medium of claim 36, wherein the call parameters include a calling party identifier, the accessing step further

including retrieving second subscriber attribute information based on the calling party identifier.

38. (canceled)

39. The medium of claim 32, wherein the accessing step includes accessing the IP-based database server according to LDAP protocol.

40. The medium of claim 32, wherein the step of accessing the IP-based messaging server includes selectively obtaining from the IP-based messaging server at least one of a subscriber name, and a subscriber greeting as a subscriber prompt based on a subscriber identifier obtained from the accessed attribute information.

41. The medium of claim 40, further comprising instructions for performing the step of converting the subscriber prompt into a media file having at least one prescribed media type.

42. The medium of claim 41, wherein the converting step includes converting the subscriber prompt into a Multipurpose Internet Media Extension (MIME) type .wav file playable by a browser.

43. The medium of claim 42, wherein the step of generating an HTML page includes inserting a first media tag including the .wav file and a second media tag configured for controlling playing of the .wav file.

44. The medium of claim 32, wherein the step of accessing the IP-based messaging server includes determining a presence of a stored message on the IP-based messaging server for the subscriber based on the subscriber messaging information, the generating step including selectively inserting one of a first prompt file specifying no new messages and a second prompt file specifying the determined presence of the stored message, based on the subscriber messaging information.

45. The medium of claim 44, wherein the step of accessing the IP-based messaging server further includes identifying, for each stored message, a corresponding message type based on the corresponding header information specifying the Multipurpose Internet Media Extension (MIME) type, the second prompt file configured for specifying the corresponding message type for each stored message.

46. The medium of claim 45, wherein the medium further comprises instructions for performing the steps of:

selecting one of the stored messages from the IP-based messaging server;

the converting step including converting the URL encoded string of the selected one message into a media file having a prescribed media type, based on the corresponding MIME type and determined capabilities of the browser having sent the first HTTP request, the generating step including inserting the media file into a media tag with a corresponding media control tag for playback of the media file by the browser.

47. The medium of claim 46, wherein the converting step includes converting the URL encoded string to text and executing a text to speech routine for converting the text into an audio file based on the header information specifying text and the determined attributes specifying audio only.

48. (canceled)

49. The medium of claim 45, wherein

the converting step includes converting selected header information into an audio file based on determining the MIME type is incompatible with determined capabilities of the browser, the generating step including inserting the audio file into the HTML page for playback by the browser.

50. (canceled)

51. The medium of claim 32, further comprising instructions for performing the steps of:

receiving a second HTTP request for storage for the subscriber of a message having a prescribed messaging format; and

outputting to the IP-based messaging server an instruction for storage of a standard-format message, containing the message and header information specifying the prescribed messaging format, in a directory specified for the subscriber.

52. The medium of claim 51, wherein the outputting step includes:

converting the message into a URL encoded string;

generating a header that specifies a Multipurpose Internet Media Extension (MIME) type for the prescribed messaging format; and

sending as the standard-format message an e-mail message, including the URL encoded string and the header as an attachment, to the IP-based messaging server according to SMTP protocol for delivery to the directory specified for the subscriber.

53. An application server configured for executing a voice messaging application, the application server including:

a hypertext transport protocol (HTTP) interface for receiving an HTTP request specifying execution of a prescribed voice messaging application operation for a subscriber; and

means for dynamically generating, in response to the HTTP request, a first hypertext markup language (HTML) document having media content for execution of the voice messaging application operation for the subscriber based on accessing attribute information for the subscriber from an Internet Protocol (IP) based database server configured for storing subscriber attributes, and based on accessing an IP-based messaging server for subscriber messaging information based on the accessed attribute information,

wherein each stored message on the IP-based messaging server is stored within a corresponding e-mail message as a URL encoded string with the corresponding header information so that each stored message is encoded in the URL encoded string.

54. The server of claim 53, wherein the generating means is configured for determining subscriber registration status and subscriber messaging preferences in response to a called party identifier specified in the HTTP request.

55. The server of claim 54, wherein the generating means includes means for accessing the IP-based database server and the IP-based messaging server according to LDAP protocol and IMAP protocol, respectively.

56. The server of claim 55, wherein the generating means is configured for accessing from the IP-based messaging server at least one of a subscriber name and a subscriber greeting as a subscriber prompt based on the HTTP request specifying a condition for a calling party to leave a message for the subscriber, the generating means converting the

subscriber prompt into a media file playable by a browser and inserting the media file into the HTML document.

57. The server of claim 56, wherein the generating means is configured for converting the subscriber prompt stored on the IP-based messaging server from the URL encoded string into an audio file as said media file.

58. The server of claim 55, wherein the generating means is configured for accessing from the IP-based messaging server a message for a subscriber, stored on the IP-based messaging server as an e-mail message having the URL encoded string and corresponding header information that specifies a Multipurpose Internet Media Extension (MIME) type, the generating means converting at least one of the header information and the URL encoded string into a media file having a selected media type based on the MIME type and determined capabilities of an input device used by the subscriber.

59. (canceled)

60. The server of claim 58, wherein the generating means is configured for converting the URL encoded string into an audio file based on the MIME type specifying a .wav type.

61. The server of claim 58, wherein the generating means is configured for converting the URL encoded string into text and converting the text into an audio file using a text to speech routine, based on the MIME type specifying text and the determined capabilities to not support text.

62. The server of claim 55, wherein the generating means is configured for converting a message supplied by the HTTP request and having a prescribed messaging format into a URL encoded string, generating a header specifying a Multipurpose Internet Media Extension (MIME) type for the prescribed messaging format, and outputting to the IP-based messaging server an e-mail message, including the URL encoded string and the header as an attachment, for delivery to a directory specified for the subscriber.

63. The server of claim 22, wherein:

the HTTP interface is configured for receiving a second HTTP request for storage for the subscriber of a message having a prescribed messaging format;

the application runtime environment configured for outputting to the IP-based messaging server an instruction for storage of a standard-format message, containing the message and header information specifying the prescribed messaging format, in a directory specified for the subscriber, based on:

(1) converting the message into the corresponding URL encoded string;

(2) generating the corresponding header that specifies the corresponding Multipurpose Internet Media Extension (MIME) type for the prescribed messaging format, and

(3) sending as the standard-format message an e-mail message, including the URL encoded string and the header as an attachment, to the IP-based messaging server according to SMTP protocol for delivery to the directory specified for the subscriber.

64. The server of claim 53, wherein:

the HTTP interface is configured for receiving a second HTTP request for storage for the subscriber of a message having a prescribed messaging format;

the generating means configured for outputting to the IP-based messaging server an instruction for storage of a standard-format message, containing the message and header information specifying the prescribed messaging format, in a directory specified for the subscriber, based on:

(1) converting the message into the corresponding URL encoded string;

(2) generating the corresponding header that specifies the corresponding Multipurpose Internet Media Extension (MIME) type for the prescribed messaging format, and

(3) sending as the standard-format message an e-mail message, including the URL encoded string and the header as an attachment, to the IP-based messaging server according to SMTP protocol for delivery to the directory specified for the subscriber.

* * * * *

65/7/5 (Item 5 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

0015145204 *Drawing available*
WPI Acc no: 2005-494777/200550
Related WPI Acc No: 2003-147216; 2003-265606; 2003-656232; 2004-313511; 2004-485761; 2004-591078; 2004-781947; 2005-119795; 2005-401610; 2005-617185; 2005-617492; 2005-699537; 2005-710219; 2006-115442; 2006-210809
XRPX Acc No: N2005-403348

Execution method of application to deliver voice portal services in application server, involves generating HTML page for execution of application, based on received HTTP request, by application instance executed by server

Patent Assignee: CISCO TECHNOLOGY INC (CISC-N)
Inventor: BARBAN S H; DODRILL L D; GARAHAN P T

Patent Family (1 patents, 1 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
US 6912691	B1	20050628	US 1999152316	P	19990903	200550	B
			US 2000608188	A	20000630		

Priority Applications (no., kind, date): US 1999152316 P 19990903; US 2000608188 A 20000630

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
US 6912691	B1	EN	11	4	Related to Provisional	US 1999152316

Alerting Abstract US B1

NOVELTY - The **attribute** information for subscriber is accessed from **database** server, on receiving a HTTP request for execution of a voice portal service **application** for subscriber. A HTML page for execution of **application** having **XML** tags for controlling delivery of the content in audible format, is generated based on request, by **application** instance executed by server. The instance is terminated after output of page to a browser.
DESCRIPTION - INDEPENDENT CLAIMS are also included for the following:

1. **application** server; and
2. **computer** readable medium storing sequences of instructions for executing **application**.

USE - For execution of **XML** voice **applications**, in **application** server (claimed), for delivery of voice portal services received through open standards-based internet protocol (IP) network e.g. world wide web (WWW), internet or corporate intranet, to clients e.g. thin client such as personal computer (PC) with web browser, skinny client such as cordless telephone, facsimile and analog telephone, fat client such as electronic **mail** (email) **application** on PC and tiny client such as advanced mobile phone service (AMPS), time division multiple access

(TDMA) or code division multiple access (CDMA)-based cellular telephone, handheld computing device e.g. 3-Com Palm computing or Windows CE-based handheld device and pager of individual or business subscribers.
ADVANTAGE - The delivery of voice portal services is deployed on platform that is customizable, scalable and built upon open standards such as IP.
DESCRIPTION OF DRAWINGS - The figure shows a schematic diagram of the voice portal service delivery system.
18a cordless telephone
18b facsimile
18c analog telephone
18d cellular telephone
18e handheld computing device

Title Terms /Index Terms/Additional Words: EXECUTE; METHOD; APPLY; DELIVER; VOICE; PORTAL; SERVICE; SERVE; GENERATE; PAGE; BASED; RECEIVE; **REQUEST**; INSTANCE

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-015/16			Main		"Version 7"
G06F-015/173			Secondary		"Version 7"

US Classification, Issued: 715513000, 709203000, 709201000

File Segment: EPI;
DWPI Class: T01; W02
Manual Codes (EPI/S-X): **T01-J18**; **T01-N01D1A**; **T01-N03A1**; **T01-N03B2**; **T01-S03**; W02-J03A7; W02-J03C8

Original Publication Data by Authority

United States

Publication No. US 6912691 B1 (Update 200550 B)
Publication Date: 20050628
Delivering voice portal services using an XML voice-enabled web server
Assignee: Cisco Technology, Inc., San Jose, CA, US (CISC-N)
Dodrill, Lewis Dean, Richmond, VA, US Residence: US
Garahan, Peter Thomas, Great Falls, VA, US Residence: US
Barban, Susan Harrow, Midlothian, VA, US Residence: US
Inventor: Dodrill, Lewis Dean, Richmond, VA, US Residence: US
Garahan, Peter Thomas, Great Falls, VA, US Residence: US

Barban, Susan Harrow, Midlothian, VA, US Residence: US
Agent: Turkevich, Leon R.
Stemberger, Edward J.
Language: EN (11 pages, 4 drawings)
Application: US 1999152316 P 19990903 (Related to Provisional)
US 2000608188 A 20000630 (Local application)
Original IPC: G06F-15/16(A) G06F-15/173(B)
Current IPC: G06F-15/16(A) G06F-15/173(B)
Original US Class (secondary): 715513 709203 709201
Original Abstract: An application server, configured for executing an application defined by XML documents for delivering voice portal services, accesses subscriber attributes from a standardized information database server (such as LDAP). The application server, upon receiving a request from a proxy browser serving a user, accesses the standardized database server to obtain attribute information for responding to the application request. The application server sends a request to a content server for media content based on the HTTP request and the attribute information. The application server generates an HTML page, for execution of the prescribed voice portal service application having XML tags configured for controlling delivery of the media content in an audible format, based on the HTTP request.
Claim:

1. 1. A method in an application server for executing an application to deliver voice portal services, the method comprising:
 - receiving a HTTP request for execution of a prescribed voice portal service application for a subscriber;
 - accessing attribute information for the subscriber from an Internet Protocol (IP) based database server configured for storing subscriber attributes;
 - sending a request to a content server for media content based on the HTTP request and the attribute information; and
 - generating an HTML page for execution of the prescribed voice portal service application having XML tags configured for controlling delivery of the media content in an audible format, based on the HTTP request,
 - wherein the generating step includes generating the HTML page by an application instance executed by the server, the method further comprising, terminating the application instance based on the HTML page having been output to a browser.

US006912691B1

(12) **United States Patent**
Dodrill et al.

(10) **Patent No.: US 6,912,691 B1**
(45) **Date of Patent: Jun. 28, 2005**

(54) **DELIVERING VOICE PORTAL SERVICES USING AN XML VOICE-ENABLED WEB SERVER**

(75) Inventors: **Lewis Dean Dodrill**, Richmond, VA (US); **Peter Thomas Garahan**, Great Falls, VA (US); **Susan Harrow Barban**, Midlothian, VA (US)

(73) Assignee: **Cisco Technology, Inc.**, San Jose, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 751 days.

(21) Appl. No.: **09/608,188**

(22) Filed: **Jun. 30, 2000**

Related U.S. Application Data

(60) Provisional application No. 60/152,316, filed on Sep. 3, 1999.

(51) **Int. Cl.**[7] **G06F 15/16**; G06F 15/173

(52) **U.S. Cl.** **715/513**; 709/203; 709/201

(58) **Field of Search** 715/513, 507, 715/517; 709/201, 203, 219, 226, 229, 250, 217; 707/5, 10; 379/100.01; 713/204, 201

(56) **References Cited**

U.S. PATENT DOCUMENTS

4,837,798 A		6/1989	Cohen et al.	
5,706,507 A	*	1/1998	Schloss	707/104.1
5,742,736 A	*	4/1998	Haddock	704/270
5,794,259 A	*	8/1998	Kikinis	715/507
5,864,870 A	*	1/1999	Guck	707/104.1
5,870,559 A	*	2/1999	Leshem et al.	709/224
5,878,417 A	*	3/1999	Baldwin et al.	707/10
5,892,825 A	*	4/1999	Mages et al.	705/51
5,892,905 A	*	4/1999	Brandt et al.	713/201
5,901,352 A	*	5/1999	St-Pierre et al.	455/426.1
6,028,917 A	*	2/2000	Creamer et al.	379/100.01
6,055,566 A	*	4/2000	Kikinis	709/219
6,173,279 B1	*	1/2001	Levin et al.	707/5
6,199,081 B1	*	3/2001	Meyerzon et al.	715/513
6,243,376 B1	*	6/2001	Ng et al.	370/352
6,269,336 B1	*	7/2001	Ladd et al.	704/270
6,501,832 B1	*	12/2002	Saylor et al.	379/88.04
6,611,862 B2	*	8/2003	Reisman	709/217
6,626,958 B1	*	9/2003	McCauley et al.	715/517

OTHER PUBLICATIONS

VoiceXML Forum, "Voice Extensible Markup Language: VoiceXML", Aug. 17, 1999, VoiceXML Forum, Version 0.9, p. 8.*

BOS, "XML in 10 Points", W3.org (Aug. 26, 1999).

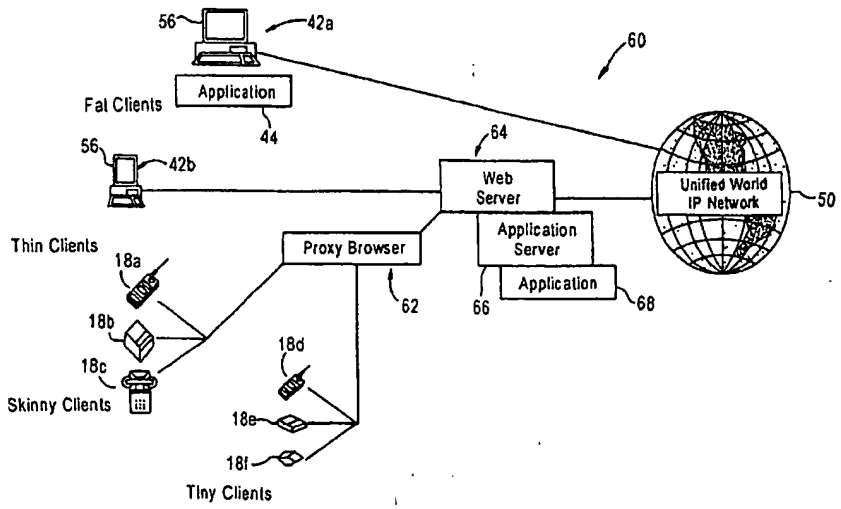
* cited by examiner

Primary Examiner—Stephen Hong
Assistant Examiner—Jonathan Schlaifer
(74) *Attorney, Agent, or Firm*—Leon R. Turkevich; Edward J. Stemberger

(57) **ABSTRACT**

An application server, configured for executing an application defined by XML documents for delivering voice portal services, accesses subscriber attributes from a standardized information database server (such as LDAP). The application server, upon receiving a request from a proxy browser serving a user, accesses the standardized database server to obtain attribute information for responding to the application request. The application server sends a request to a content server for media content based on the HTTP request and the attribute information. The application server generates an HTML page, for execution of the prescribed voice portal service application having XML tags configured for controlling delivery of the media content in an audible format, based on the HTTP request.

38 Claims, 4 Drawing Sheets



example, the PHP hypertext processor includes a high-speed interface for Apache Web servers.

The application server **66** also includes a runtime environment **100** for execution of the parsed XML documents. As described above, the runtime environment **100** may selectively execute any one of user interface operation **98**, a logic operation **102**, or a procedure call **104** as specified by the parsed XML document by executing a corresponding set of executable functions based on the rule set for the corresponding operation. In particular, the application runtime environment **100** includes a tag implementation module **110** that implements the XML tags parsed by the XML parser **94**. The tag implementation module **110** performs relatively low-level operations, for example dynamically generating an XML menu page using executable functions specified by a menu rule set in response to detecting a menu tag, performing a logical operation using executable functions specified by a logic rule set in response to a decision tag, or fetching an audio (.wav) file in response to detecting a sound tag. Hence, the tag implementation module **110** implements the tag operations that are specified within the XML framework of the stored XML documents.

The application server **66** also includes a set of libraries **82, 83** that may be implemented as dynamically linked libraries (DLLs) or application programming interface (API) libraries. The libraries **82** and **83** enable the runtime environment **100** to implement the procedures **104** as specified by the appropriate XML document. The application server **66** may issue a function call to one of a plurality of IP protocol compliant remote resources **86, 88, 91,** or **93**. Note that the mechanisms for accessing the services **86, 88, 91,** or **93** should be established within the application server before use of XML documents that reference those services. Once the services **86, 88, 91,** or **93** are established, the application runtime environment **100** can perform a function operation by using executable functions specified by a function call rule set.

The arrangement for delivering voice portal services will now be described.

FIG. **4** is a diagram illustrating delivery of voice portal services executed by the application server **66** according to an embodiment of the present invention. The steps described in FIG. **4** can be implemented as executable code stored on a computer readable medium (e.g., a hard disk drive, a floppy drive, a random access memory, a read only memory, an EPROM, a compact disc, etc.).

The application server **66** accesses news, weather, sports, stock, etc, information using the data retrieval API **83** to obtain content information from the steaming information server **93**, XML pages or structured content **87** or the HTML web pages via the web server **91**. The application server **66** is configured for delivering web portal services by obtaining subscriber information from the LDAP directory **84**. For example, the subscriber information may include subscriber device capabilities and subscriber preferences such as a personalized menu of subjects for which the subscriber wishes to receive information. For example, subscriber information may include subscriber preferences for subscriber information services such as ""My Yahoo", normally stored for a browser using a cookie; in this case, the cookie may be stored in the LDAP directory **84** (or generated by the application server **66**), causing the API **83** to output an HTTP GET request having the cookie for retrieval of the personalized content from the content server **91**.

As illustrated in FIG. **4**, the application server **66** receives in step **240** an HTTP request from the proxy browser **62** serving the subscriber. In response, the application server **66** accesses an XML document **68** from the database **72** in step **242**, and generates a new brownie or accesses an existing brownie in step **244** to determine what web content the subscriber wishes to receive. The application server **66** then accesses the subscriber attribute information in step **246** from the LDAP directory **84**, for example subscriber identity, subscriber status, preferences, and device capabilities. The application server **66** also retrieves from the web server **91**, based on the subscriber attribute information obtained in step **246**, the XML pages/structured content **87**, or the HTML pages **95** through the API **83** in step **248**. The application server **66** also may retrieve streaming information (e.g., a URL) from the information server **93** to enable the browser **62** to initiate transmission of a media stream from the server **93**. The application server **66** adds media content such as audio (.wav) files (or URL's that point to the audio files) to the accessed web pages containing the information and/or converts accessed text based information into audio files using the text-to-speech resource **88** in step **254**. The application server **66** then generates an HTML document having XML tags that specify parameters of reception of the media stream and the proxy browser **62** plays the media stream for the subscriber. For non-streaming information, the application server generates an HTML document having XML tags that specify playing of the audio files in step **256** and forwards the HTML document to the proxy browser **62** for playing the files to the subscriber. The application server **66** then updates and stores the brownie in step **258**.

Hence, the subscriber can receive, via a telephone **18**, voice portal services. More detailed information can be obtained by use of menus such as, "To receive stock quotes, press 1, to receive headline news, press 2" and so on. Once the input is made, the application server **66** can generate a second HTML document having instructions to retrieve the detailed information requested. Further menus can be used by the subscriber to gain even more detailed information such as the price of a particular stock.

The XML document **68** can include the subscriber's telephone number so that, for example, if a stock of interest reaches a certain price, the application server **66** can generate another HTML document with tags to instruct the proxy browser **62** to cause a telephony device to call or page the subscriber.

According to the disclosed embodiment, delivery of voice portal services may be deployed on a platform that is customizable, scalable, and built upon open standards such as Internet protocol. Users can access portal services via PC-based browser or via a telephone and proxy browser.

While this invention has been described in connection with what is presently considered to be the most practical and preferred embodiment, it is to be understood that the invention is not limited to the disclosed embodiments, but, on the contrary, is intended to cover various modifications and equivalent arrangements included within the spirit and scope of the appended claims.

What is claimed is:

1. A method in an application server for executing an application to deliver voice portal services, the method comprising:

receiving a HTTP request for execution of a prescribed voice portal service application for a subscriber;

accessing attribute information for the subscriber from an Internet Protocol (IP) based database server configured for storing subscriber attributes;

sending a request to a content server for media content based on the HTTP request and the attribute information; and

generating an HTML page for execution of the prescribed voice portal service application having XML tags configured for controlling delivery of the media content in an audible format, based on the HTTP request,

wherein the generating step includes generating the HTML page by an application instance executed by the server, the method further comprising, terminating the application instance based on the HTML page having been output to a browser.

2. The method of claim 1, wherein the receiving step includes recovering within the HTTP request a browser configuration.

3. The method of claim 2, wherein the recovering step includes identifying the browser configuration as one of a computer browser configuration configured for parsing a prescribed group of media tags and presenting a prescribed group of media types, and a lightweight browser configuration configured for parsing a prescribed portion of the prescribed group of media tags.

4. The method of claim 3, wherein the generating step includes generating the HTML page by selectively supplying media tag types based on the identified browser configuration.

5. The method of claim 1, wherein the accessing step includes accessing the IP-based database server according to the LDAP protocol.

6. The method of claim 1, wherein the step of sending a request includes using a data retrieval application programming interface to access the content server.

7. The method of claim 6, further comprising converting text based information obtained from the content server into a media file having at least one prescribed media type.

8. The method of claim 7, wherein the converting step includes converting the text based information into a .wav file playable by a browser.

9. The method of claim 8, wherein the step of generating an HTML page includes inserting a first media tag including the .wav file and a second media tag configured for playing the .wav file.

10. The method of claim 7, wherein the converting step includes executing a text to speech resource for converting the text based information into an audio file.

11. The method of claim 1, further comprising generating a second HTML page in response to an input indicating particular information of interest received from the subscriber, having instructions to access the particular information.

12. The method of claim 1, wherein the step of accessing attribute information includes accessing subscriber preferences defining information which the subscriber wishes to receive.

13. An application server configured for executing an application for delivering voice portal services, the application server including:

a hypertext transport protocol (HTTP) interface for receiving an HTTP request specifying execution of a prescribed voice portal application for a subscriber; and

an application runtime environment configured for dynamically generating, in response to the HTTP request, a hypertext markup language (HTML) document for execution of the prescribed voice portal service application, the HTML document having XML tags configured for controlling delivery of media content, from a content server, in an audible format based on the HTTP request and on accessing attribute information for the subscriber from an Internet Protocol (IP) based database server configured for storing subscriber attributes,

wherein the application runtime environment is configured for generating the HTML document by an application instance, wherein the application instance is terminated based on the HTML document having been output to a browser.

14. The server of claim 13, wherein the application runtime environment accesses the IP-based database server according to the LDAP protocol.

15. The server of claim 13, wherein the application runtime environment is configured for instructing a data retrieval application programming interface to access the content server.

16. The server of claim 13, wherein the application runtime environment is configured for converting text based information, obtained from the content server, into a media file having at least one prescribed media type.

17. The server of claim 16, wherein the application runtime environment is configured for converting the text based information into a .wav file playable by a browser.

18. The server of claim 16, wherein application runtime environment is configured for executing a text to speech resource for converting the text based information into an audio file.

19. The server of claim 13, wherein the attribute information includes subscriber preferences defining information which the subscriber wishes to receive.

20. A computer readable medium having stored thereon sequences of instructions for executing an application to deliver voice portal services, the sequences of instructions including instructions for performing the steps of:

receiving a HTTP request for execution of a prescribed voice portal service application for a subscriber;

accessing attribute information for the subscriber from an Internet Protocol (IP) based database server configured for storing subscriber attributes;

sending a request to a content server for media content based on the HTTP request and the attribute information; and

generating an HTML page for execution of the prescribed voice portal service application having XML tags configured for controlling delivery of the media content in an audible format, based on the HTTP request,

wherein the generating step includes generating the HTML page by an application instance executed by the server, the sequence of instructions further including instructions for terminating the application instance based on the HTML page having been output to a browser.

21. The medium of claim 20, wherein the receiving step includes recovering within the HTTP request a browser configuration.

22. The medium of claim 21, wherein the recovering step includes identifying the browser configuration as one of a computer browser configuration configured for parsing a prescribed group of media tags and presenting a prescribed group of media types, and a lightweight browser configuration configured for parsing a prescribed portion of the prescribed group of media tags.

23. The medium of claim 21, wherein the generating step includes generating the HTML page by selectively supplying media tag types based on the identified browser configuration.

24. The medium of claim 20, wherein the accessing step includes accessing the IP-based database server according to the LDAP protocol.

25. The medium of claim 20, wherein the step of sending a request includes using a data retrieval application programming interface to access the content server.

26. The medium of claim 25, further comprising converting text based information obtained from the content server, into a media file having at least one prescribed media type.

27. The medium of claim 26, wherein the converting step includes converting the text based information into a .wav file playable by a browser.

28. The medium of claim 26, wherein the step of generating an HTML page includes inserting a first media tag including the .wav file and a second media tag configured for playing the .wav file.

29. The medium of claim 25, wherein the converting step includes executing a text to speech resource for converting the text based information into an audio file.

30. The medium of claim 20, further comprising generating second HTML page in response to an input indicating particular information of interest received from the subscriber, having instructions to access the particular information.

31. The medium of claim 20, wherein the step of accessing attribute information includes accessing subscriber preferences defining information which the subscriber wishes to receive.

32. An application server configured for executing an application for delivering a voice portal service, the application server including:

a hypertext transport protocol (HTTP) interface for receiving an HTTP request specifying execution of a prescribed voice portal application for a subscriber; and

means for dynamically generating, in response to the HTTP request, a hypertext markup language (HTML) document for execution of the prescribed voice portal service application, the HTML document having XML tags configured for controlling delivery of media content, from a content server, in an audible format based on the HTTP request and on accessing attribute information for the subscriber from an Internet Protocol (IP) based database server configured for storing subscriber attributes,

wherein the generating means is configured for generating the HTML document by an application instance, wherein the application instance is terminated based on the HTML document having been output to a browser.

33. The server of claim 32, wherein the generating means is configured to access the IP based database server according to the LDAP protocol.

34. The server of claim 32, wherein the generating means is configured to instruct a data retrieval application programming interface to access the content server.

35. The server of claim 32, wherein the generating means is configured for converting text based information into a media file having at least one prescribed media type.

36. The server of claim 35, wherein the generating means is configured for converting the text based information into a .wav file playable by a browser.

37. The server of claim 35, wherein generating means is configured for executing a text to speech resource for converting the text based information into an audio file.

38. The server of claim 32, wherein the attribute information includes subscriber preferences defining information which the subscriber wishes to receive.

* * * * *

65/7/36 (Item 36 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

0011022760 *Drawing available*
WPI Acc no: 2001-648475/200174
XRAM Acc no: C2001-191379
XRPX Acc No: N2001-484539
A new database management system comprises linking databases of genetic sequences, chemical structure information and biological assays displayed as a web page and useful for storing and retrieving biomolecular data
Patent Assignee: ARENA PHARM INC (AREN-N)
Inventor: JONES G

Patent Family (3 patents, 92 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
WO 2001073587	A2	20011004	WO 2001US3038	A	20010130	200174	B
AU 200133134	A	20011008	AU 200133134	A	20010130	200208	E
AU 2001233134	A8	20051006	AU 2001233134	A	20010130	200612	E

Priority Applications (no., kind, date): US 2000193065 P 20000329; US 2000635833 A 20000809

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
WO 2001073587	A2	EN	41	23		
National Designated States,Original	AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CR CU CZ DE DK DM DZ EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM TR TT TZ UA UG US UZ VN YU ZA ZW					
Regional Designated States,Original	AT BE CH CY DE DK EA ES FI FR GB GH GM GR IE IT KE LS LU MC MW MZ NL OA PT SD SE SL SZ TR TZ UG ZW					
AU 200133134	A	EN			Based on OPI patent	WO 2001073587
AU 2001233134	A8	EN			Based on OPI patent	WO 2001073587

Alerting Abstract WO A2
NOVELTY - A **database** management system providing an **interface** to **databases** storing biomolecular **data**, comprising the **databases**, a processing unit, computer instructions for accessing the **databases** and receiving the **data**, and a web page generated by the processing unit which displays the **data** retrieved from the **databases**, is new.
USE - The system is used for storing, using, maintaining and retrieving large amounts of biomolecular **data** including chemical, biological, genomic, testing and other related **data** and information.

ADVANTAGE - Unlike prior art the invention allows real time linking of gathered **data** with search **data**, linking of biological assay **data** to chemical structure information and gene sequence information
DESCRIPTION OF DRAWINGS - An overview of one embodiment of the system.

Technology Focus
BIOTECHNOLOGY - Preferred System: Biomolecular **data** is preferably genomic, chemical structure or gene sequences. The **databases** preferably include a **relational database** that stores statistical biomolecular **data**, particularly a molecule identification number, structural descriptor or assay results. The **databases** also preferably comprise a chemical structure search system, preferably a UNITY system, and searches may be for similar or exact match molecules. A bioinformatics **database** may also be included. Gene sequences may be held and searched for homology. A **data** gathering instrument is also preferably part of the system. The web **page** may be in HTML or **XML** format and comprise a histogram, table or chemical structure and preferably comprises an **applet** which updates **search** progress made. The **applet** may be **generated** by generating a colored grid displaying visualization of plate results or visualization of dose response **data**. Preferably the system comprises a system of networked computers more preferably the Internet. The figure provides one example of how the system may be set up.

Title Terms /Index Terms/Additional Words: NEW; **DATABASE**; MANAGEMENT; SYSTEM; COMPRISE; LINK; GENETIC; SEQUENCE; CHEMICAL; STRUCTURE; INFORMATION; BIOLOGICAL; ASSAY; DISPLAY; WEB; PAGE; USEFUL; STORAGE; RETRIEVAL; **DATA**

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-017/00; G06F-017/30			Main		"Version 7"

File Segment: CPI; EPI
DWPI Class: B04; D16; T01
Manual Codes (EPI/S-X): **T01-J**
Manual Codes (CPI/A-N): B04-E01; B11-C; B11-C08; B11-C09; D05-H09; D05-H12; D05-H18

Original Publication Data by Authority

Australia

Publication No. AU 2001233134 A8 (Update 200612 E)
Publication Date: 20051006
Universal biomolecular data system
Assignee: ARENA PHARM INC (AREN-N)
Inventor: JONES G
Language: EN

Application: AU 2001233134 A 20010130 (Local application)
Priority: US 2000193065 P 20000329
US 2000635833 A 20000809
Related Publication: WO 2001073587 A (Based on OPI patent)
Original IPC: G06F-17/30(A)
Current IPC: G06F-17/30(A)

Publication No. AU 200133134 A (Update 200208 E)
Publication Date: 20011008
Assignee: ARENA PHARM INC; US (AREN-N)
Language: EN
Application: AU 200133134 A 20010130 (Local application)
Priority: US 2000193065 P 20000329
US 2000635833 A 20000809
Related Publication: WO 2001073587 A (Based on OPI patent)

WIPO

Publication No. WO 2001073587 A2 (Update 200174 B)
Publication Date: 20011004
UNIVERSAL BIOMOLECULAR DATA SYSTEM
SYSTEME UNIVERSEL DE DONNEES BIOMOLECULAIRES
Assignee: *(except US)* ARENA PHARMACEUTICALS, INC., 6166 Nancy Ridge Drive, San Diego, CA 92121, US Residence: US Nationality: US (AREN-N)
(only US) JONES, Gareth, 210 South Helix Avenue, Apt. H, Solana Beach, CA 92075, US Residence: US Nationality: GB
Inventor: JONES, Gareth, 210 South Helix Avenue, Apt. H, Solana Beach, CA 92075, US Residence: US Nationality: GB
Agent: SKALE, Andrew, Brobeck, Phleger & Harrison, LLP, 12390 El Camino Real, San Diego, CA 92130, US
Language: EN (41 pages, 23 drawings)
Application: WO 2001US3038 A 20010130 (Local application)
Priority: US 2000193065 P 20000329
US 2000635833 A 20000809
Designated States: (National Original) AE AG AL AM AT AU AZ BA BB BG BR BY BZ CA CH CN CR CU CZ DE DK DM DZ EE ES FI GB GD GE GH GM HR HU ID IL IN IS JP KE KG KP KR KZ LC LK LR LS LT LU LV MA MD MG MK MN MW MX MZ NO NZ PL PT RO RU SD SE SG SI SK SL TJ TM TR TT TZ UA UG US UZ VN YU ZA ZW
(Regional Original) AT BE CH CY DE DK EA ES FI FR GB GH GM GR IE IT KE LS LU MC MW MZ NL OA PT SD SE SL SZ TR TZ UG ZW
Original IPC: G06F-17/00(A)
Current IPC: G06F-17/00(A)
Original Abstract: The present invention includes a database management system that interfaces with a plurality of databases storing biomolecular data. A request to access biomolecular data stored in at least one of a plurality of databases is received by the database management system from a user's computer. The database management system determines which of the plurality of databases store the biomolecular data. Instructions to access the biomolecular data in the at least one of the plurality of databases is generated and the information is accessed. The biomolecular

data is received from the at least one of the plurality of databases. Then a web page display of the biomolecular data received from the at least one of the plurality of databases is generated and sent to the user's computer.

65/7/34 (Item 34 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

0011191903 *Drawing available*
WPI Acc no: 2002-130098/200217
XRPX Acc No: N2002-098143
Computerized interface mapping system in software development field, performs mapping of methods, properties and events of ActiveX compatible interface and JavaBean compatible interface
Patent Assignee: MICROSOFT CORP (MICT); SHARPE T C (SHAR-I); TOUTONGHI M J (TOUT-I)
Inventor: SHARPE T C; TOUTONGHI M J

Patent Family (2 patents, 1 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
US 20020004934	A1	20020110	US 1998115987	A	19980715	200217	B
US 6438744	B1	20020820	US 1998115987	A	19980715	200257	E

Priority Applications (no., kind, date): US 1998115987 A 19980715

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
US 20020004934	A1	EN	12	3		

Alerting Abstract US A1
NOVELTY - A Microsoft window operating system includes ActiveX compatible **interface** and a **JavaBean** compatible **interface** for controlling respective **software** components. A client **program** with a run-time, utilizes one **interface** with the **software** component of the other **interface**. The methods, properties and events of the **interfaces** are **mapped**, during the run-time of the client **program**.
DESCRIPTION - INDEPENDENT CLAIMS are also included for the following:

A. Computer-readable medium storing **interface mapping program**;
B. Computer;
C. Computerized interface mapping method

USE - For dynamic **mapping** of **software** component **interface** e.g. ActiveX compatible **interface** used by **applications** written in Microsoft's Visual Basic and Visual C++ programming languages, **JavaBean** compatible **interface**, etc, in **software** development field.
ADVANTAGE - Dynamic **mapping** does not require the use of extraneous files that must be administered and maintained, and is automatically handled by virtual machines. Allows **software** developers to **map** between component models defined in different development environments that can provide for a more complete **mapping** of component objects and information within the object, while reducing the development and maintenance overhead of current **mapping** techniques.

DESCRIPTION OF DRAWINGS - The figure shows the flowchart explaining the computerized method for dynamic **mapping** of **interfaces**.

Title Terms /Index Terms/Additional Words: **INTERFACE**; **MAP**; SYSTEM; **SOFTWARE**; DEVELOP; FIELD; PERFORMANCE; METHOD; PROPERTIES; EVENT; COMPATIBLE

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-009/45			Main		"Version 7"

US Classification, Issued: 717005000, 717137000, 717106000, 717107000, 717108000, 717116000, 717117000, 717118000, 717162000, 717163000, 717164000 , 717165000

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): **T01-J20B1**; **T01-S03**

Original Publication Data by Authority

United States

Publication No. US 20020004934 A1 (Update 200217 B)
Publication Date: 20020110
DYNAMIC MAPPING OF COMPONENT INTERFACES
Assignee: TOUTONGHI, MICHAEL J., SEATTLE, WA, US (TOUT-I)
SHARPE, TRACY C., SEATTLE, WA, US (SHAR-I)
Inventor: TOUTONGHI, MICHAEL J., SEATTLE, WA, US
SHARPE, TRACY C., SEATTLE, WA, US
Agent: KLARQUIST SPARKMAN CAMPBELL LEIGH & WHINSTON LLP, 121 S.W. SALMON STREET, SUITE 1600, PORTLAND, OR, US
Language: EN (12 pages, 3 drawings)
Application: US 1998115987 A 19980715 (Local application)
Original IPC: G06F-9/45(A)
Current IPC: G06F-9/45(A)
Original US Class (main): 7175
Original Abstract: The dynamic mapping from an ActiveX component model to a JavaBean model is disclosed. In one embodiment, an ActiveX compatible object is created at run-time for those JavaBean components that an ActiveX client application wishes to utilize. In another embodiment, upon a call to CoGetClassObject or related methods using class identifiers, a COM-callable wrapper is generated and a mapping table containing COM dispatch identifiers is scanned to ensure the dispatch identifiers are unique. If they are not, the dispatch identifiers associated with JavaBean component interfaces are adjusted to make them unique.
Claim: We claim:

1. 1. A computerized system comprising:
 - an operating system providing a first interface for controlling a first component and a second interface for controlling a second component, said first interface and said second interface each having methods, properties and events;
 - at least one client program utilizing said first interface and said second component, said client program having a run-time; and
 - a mapping from the methods, properties and events of said first interface to the methods, properties and events of said second interface, said mapping occurring during the run-time of the client program.

Publication No. US 6438744 B1 (Update 200257 E)
Publication Date: 20020820
Dynamic mapping of component interfaces.
Assignee: Microsoft Corporation, Redmond, WA, US (MICT)
Inventor: Toutonghi, Michael J., Seattle, WA, US
Sharpe, Tracy C., Seattle, WA, US
Agent: Klarquist Sparkman, LLP
Language: EN
Application: US 1998115987 A 19980715 (Local application)
Original IPC: G06F-9/45(A)
Current IPC: G06F-9/45(A)
Original US Class (main): 717137
Original US Class (secondary): 717106 717107 717108 717116 717117 717118 717162 717163 717164 717165
Original Abstract: The dynamic mapping from an ActiveX component model to a JavaBean model is disclosed. In one embodiment, an ActiveX compatible object is created at run-time for those JavaBean components that an ActiveX client application wishes to utilize. In another embodiment, upon a call to CoGetClassObject or related methods using class identifiers, a COM-callable wrapper is generated and a mapping table containing COM dispatch identifiers is scanned to ensure the dispatch identifiers are unique. If they are not, the dispatch identifiers associated with JavaBean component interfaces are adjusted to make them unique.
Claim:

1. A computerize system comprising:
 - an operating system providing a first interface structured according to a first object model for controlling a first component structured according to the first object model and a second interface structured according to a second object model for controlling a second component structured according to the second object model, said first interface and said second interface each having methods and properties;
 - at least one client program utilizing said first interface to invoke a first method on the first component at run-time but instead invoking a second method on the second componet;
 - a runtime queriable component defined in the second object model; and
 - a data structure representing a mapping from methods and properties of said first interface to methods and properties of said second interface, said mapping data structure generated at least in part with a query made on the runtime queriable component;
 - wherein the first object model is different from the second object model.

US 20020004934A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2002/0004934 A1

TOUTONGHI et al. (43) Pub. Date: Jan. 10, 2002

(54) DYNAMIC MAPPING OF COMPONENT INTERFACES

(76) Inventors: MICHAEL J. TOUTONGHI, SEATTLE, WA (US); TRACY C. SHARPE, SEATTLE, WA (US)

Correspondence Address:
KLARQUIST SPARKMAN CAMPBELL
LEIGH & WHINSTON LLP
121 S.W. SALMON STREET
SUITE 1600
PORTLAND, OR 97204 (US)

(*) Notice: This is a publication of a continued prosecution application (CPA) filed under 37 CFR 1.53(d).

(21) Appl. No.: 09/115,987

(22) Filed: Jul. 15, 1998

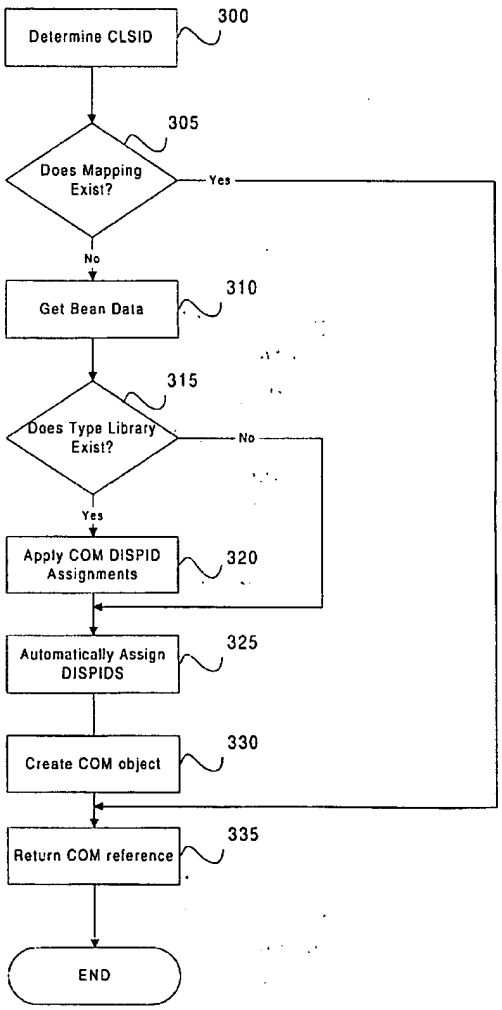
Publication Classification

(51) Int. Cl.[7] G06F 9/45

(52) U.S. Cl. 717/5

(57) ABSTRACT

The dynamic mapping from an ActiveX component model to a JavaBean model is disclosed. In one embodiment, an ActiveX compatible object is created at run-time for those JavaBean components that an ActiveX client application wishes to utilize. In another embodiment, upon a call to CoGetClassObject or related methods using class identifiers, a COM-callable wrapper is generated and a mapping table containing COM dispatch identifiers is scanned to ensure the dispatch identifiers are unique. If they are not, the dispatch identifiers associated with JavaBean component interfaces are adjusted to make them unique.



the CLSID. Preferably, the type library contains information about components and their interfaces, methods, properties, method arguments, and structures. If a type library does exist for the class, **320** applies any COM dispatch identifier (DISPID) assignments from the type library to the runtime mapping of DISPIDs to JavaBeans methods, properties and events. In one embodiment of the invention, the mapped DISPIDs are checked to make sure they do not conflict with commonly used DISPIDs. If there are conflicts, the mapped DISPIDs are adjusted to make them unique. This is generally necessary because some ActiveX clients have predefined expectations about the functionality provided by certain DISPID values and were not designed to be flexible.

[0044] At **325**, an automatic assignment of DISPIDs to JavaBeans methods, properties and events is performed. In one embodiment of the invention, this assignment is based on the naming convention for a bean's methods, properties and events, and the functional description obtained at runtime from the type library.

[0045] Finally, at **335** a reference to a COM object is returned to the caller of the method and the method stops. If the check at **305** determined that a dynamic mapping did not already exist, the reference is to the object created at **330**. Otherwise, the reference returned is to an object that was created in a previous dynamic mapping for the CLSID. Re-using the previously generated COM object is desirable because less memory and CPU resources are used.

Alternative Method of an Exemplary Embodiment for Dynamically Mapping a JavaBean to a COM Object

[0046] The previous section described the general case where a COM method is invoked specifying a component that is implemented by a Java bean. This section describes processing that must take place when an ActiveX client calls COM methods that create component instances using a class identifier (CLSID), such as "CoGetClassObject", "CoCreateInstance", or other related APIs for a component that is implemented as a Java bean. Because the client only needs to know the CLSID, the client may be unaware that the component is implemented as a bean. The method is similar to that described in the previous section, with two differences. First, the CLSID is already known and does not have to be determined. Second, the method does not need to apply COM DISPID assignments from the type library.

[0047] Referring now to **FIG. 3(*b*)**, a flowchart for a computer implemented method to dynamically map a bean invoked as a result of a COM client call to CoGetClassObject and related APIs is presented. The method begins at **350** by checking if a dynamic mapping already exists to the Java class. If not, **355** obtains data on the bean's methods, properties and events using the "java.beans.Introspector." The method proceeds to **360** which applies an automatic assignment of DISPIDs to JavaBean methods, properties and events. These values are dynamically mapped in the same manner as described with reference to **FIG. 3(*a*)**.

[0048] Next, **370** creates a COM object that implements conventional ItypeLib and ItypeInfo interfaces using and wraps the data gathered at **355** and **365**.

[0049] Finally, **375** returns a reference to a COM object. If the check at **350** determined a mapping did not already exist, the reference returned is to the object created at **370**. Otherwise, the reference returned is to the previously created object.

Conclusion

[0050] The dynamic mapping of an ActiveX control to a Java bean has been described. In particular, systems and methods of creating a mapping using information gathered at run-time from a Java bean component have been described. Although specific embodiments have been illustrated and described herein, it will be appreciated by those of ordinary skill in the art that any arrangement which is calculated to achieve the same purpose may be substituted for the specific embodiments shown. For example, the systems and methods described herein could be applied to component model interfaces other than ActiveX or JavaBeans. This application is intended to cover any adaptations or variations of the present invention. Therefore, it is manifestly intended that this invention be limited only by the following claims and equivalents thereof.

We claim:

1. A computerized system comprising:

an operating system providing a first interface for controlling a first component and a second interface for controlling a second component, said first interface and said second interface each having methods, properties and events;

at least one client program utilizing said first interface and said second component, said client program having a run-time; and

a mapping from the methods, properties and events of said first interface to the methods, properties and events of said second interface, said mapping occurring during the run-time of the client program.

2. The computerized system of claim 1, wherein the first interface comprises an ActiveX compatible interface.

3. The computerized system of claim 1, wherein the second interface comprises a JavaBean compatible interface.

4. The computerized system of claim 1 wherein the second component is a Java bean.

5. The computerized system of claim 1, wherein the operating system is a version of the Microsoft Windows operating system.

6. A computerized method for dynamically mapping a first component having an interface comprising methods, properties and events to a second component having an interface comprising methods, properties and events, the computerized method comprising:

generating an object representing said first component;

querying said second component for its methods properties and events to create a set of source data; and

populating said object with said source data.

7. The computerized method of claim 6, wherein said object is a COM-callable object.

8. The computerized method of claim 6, wherein said second component is a Java bean.

9. The computerized method of claim 6, further comprising:

generating a data structure containing a plurality of entries representing the mapping between said first component

interface and said second component interface, said entries each containing a Dispatch Identifier (DISPID); and

adjusting said Dispatch Identifier to eliminate conflicts within the Dispatch Identifiers in the plurality of entries.

10. A computer-readable medium having computer-executable instructions for performing a method comprising the steps of:

defining a first component having an interface comprising methods, properties and events;

defining a second component having an interface comprising methods, properties and events;

generating an object representing said first component;

querying said second component for its methods properties and events to create a set of source data; and

populating said object with said source data.

11. The computer-readable medium of claim 10 wherein the step of generating an object comprises the step of generating an ActiveX compatible object.

12. The computer-readable medium of claim 10 wherein the step of defining a second component comprises the step of defining a Java bean.

13. The computer-readable medium of claim 10 further having computer-executable instructions for performing the steps of:

generating a data structure containing a plurality of entries representing the mapping between said first component interface and said second component interface, said entries each containing a Dispatch Identifier (DISPID); and

adjusting said Dispatch Identifier to eliminate conflicts within the Dispatch Identifiers in the plurality of entries.

14. A computer comprising:

a processor;

a computer-readable medium; and

an operating system executed by the processor from the medium and providing:

a first interface for controlling a first component and a second interface for controlling a second component, said first interface and said second interface each having methods, properties and events;

at least one client program utilizing said first interface and said second component, said client program having a run-time; and

a mapping from the methods, properties and events of said first interface to the methods, properties and events of said second interface, said mapping occurring during the run-time of the client program.

15. The computer of claim 13, wherein the computer-readable medium comprises a memory.

16. The computer of claim 13, wherein the first interface comprises an ActiveX compatible interface.

17. The computer of claim 13, wherein the second interface comprises a JavaBean compatible interface.

18. The computer of claim 13, wherein the operating system is a version of the Microsoft Windows operating system.

* * * * *

65/7/1 (Item 1 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

SAME APPLICANT

0015881254 *Drawing available*
WPI Acc no: 2006-412931/200642
XRPX Acc No: N2006-341929

Query execution method in relational database management system, involves retrieving meta data from result set by executing modified query

Patent Assignee: INT BUSINESS MACHINES CORP (IBMC)
Inventor: MANDEL R H

Patent Family (1 patents, 1 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
US 7058620	B1	20060606	US 2000545592	A	20000407	200642	B

Priority Applications (no., kind, date): US 2000545592 A 20000407

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
US 7058620	B1	EN	15	6		

Alerting Abstract US B1
NOVELTY - The query is modified to replace selected clauses with false clauses. A **meta data** is retrieved from a result set by **executing** the modified **query**.
DESCRIPTION - INDEPENDENT CLAIMS are also included for the following:

1. apparatus for **executing** a **query**; and
2. article of manufacture comprising a **computer program** carrier for executing a query against a **database** on a **data** storage device **connected** to the computer.

USE - For **relational database** management system (**RDBMS**).
ADVANTAGE - **Meta data** is extracted across different platforms.
DESCRIPTION OF DRAWINGS - The figure shows the flowchart explaining the steps involved in extracting **meta data**.

Title Terms /Index Terms/Additional Words: QUERY; EXECUTE; METHOD; RELATED; **DATABASE**; MANAGEMENT; SYSTEM; RETRIEVAL; META; **DATA**; RESULT; SET; MODIFIED

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date

G06F-0007/00	A	I	F	B	20060101

US Classification, Issued: 707002000, 707001000, 707003000, 707004000, 707005000

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): **T01-J05B3**; **T01-J05B4B**; **T01-J05B4M**; **T01-S03**

Original Publication Data by Authority

United States

Publication No. US 7058620 B1 (Update 200642 B)
Publication Date: 20060606
Cross-platform subselect metadata extraction
Assignee: International Business Machines Corporation, Armonk, NY, US (IBMC)
Mandel, III, Richard Henry, San Jose, CA, US Residence: US
Inventor: Mandel, III, Richard Henry, San Jose, CA, US Residence: US
Agent: Sughrue Mion, PLLC
Language: EN (15 pages, 6 drawings)
Application: US 2000545592 A 20000407 (Local application)
Original IPC: G06F-7/00(B,I,H,US,20060101,20060606,A,F)
Current IPC: G06F-7/00(B,I,H,US,20060101,20060606,A,F)
Original US Class (secondary): 7072 7071 7073 7074 7075
Original Abstract: A metadata extraction system executes a query against a database on a data storage device connected to a computer. Initially, the query is modified to replace one or more selected clauses with a false clause. The modified query with the false clause is executed. Metadata is retrieved from the result set obtained by executing the modified query. The metadata is used to obtain column types, which are converted to Java(TM) types. Then, a SQLJ iterator is generated, which has parameters for the Java(TM) types.
Claim: What is claimed is:

1. 1. A method for executing a query against a database on a data storage device connected to a computer, the method comprising:
 - modifying the query to replace one or more selected clauses with one or more false clauses;
 - executing the modified query with the one or more false clauses; and
 - retrieving metadata from the result set obtained by executing the modified query.

65/7/15 (Item 15 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

SAME APPLICANT

0013276048 *Drawing available*
WPI Acc no: 2003-362144/200334
Related WPI Acc No: 2002-731552; 2003-039869; 2003-139205
XRPX Acc No: N2003-289189

Data location method in computer database system, involves storing document in main table, documents elements/ attributes in side tables and using side tables to locate data in main table

Patent Assignee: INT BUSINESS MACHINES CORP (IBMC)
Inventor: CHAU H K; CHENG I K; CHENG J M; CHIU S M; CHOW J; PAUSER M L; XU J

Patent Family (1 patents, 1 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
US 20030014397	A1	20030116	US 1999168659	P	19991202	200334	B
			US 2000725363	A	20001129		
			US 200262127	A	20020131		

Priority Applications (no., kind, date): US 2000725363 A 20001129; US 1999168659 P 19991202; US 200262127 A 20020131

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
US 20030014397	A1	EN	79	11	Related to Provisional	US 1999168659
					Continuation of application	US 2000725363

Alerting Abstract US A1
NOVELTY - A main table (300) is created and stored with a document in its column. Side tables (302,304,306,308) are created and stored with the elements or **attributes** of the document, and a **data** in the main table is **located** using the side tables.
DESCRIPTION - INDEPENDENT CLAIMS are included for the following:

1. **data** location apparatus;
2. article of manufacture comprising recorded medium storing **data locating program**;
3. **data** transforming method;
4. **data** transforming apparatus; and
5. article of manufacture comprising recorded medium storing **data** transforming **program**.

USE - In computer **database** systems.
ADVANTAGE - Enables to easily and quickly **locate data** stored in the main table.
DESCRIPTION OF DRAWINGS - The figure shows an illustration of an **application** of a main table and its four

side tables.
300 main table
302,304,306,308 side tables

Title Terms /Index Terms/Additional Words: **DATA**; **LOCATE**; METHOD; COMPUTER; **DATABASE**; SYSTEM; STORAGE; DOCUMENT ; MAIN; TABLE; ELEMENT; **ATTRIBUTE**; SIDE

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-007/00			Main		"Version-7"

US Classification, Issued: 707003000

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): **T01-J05B1**; **T01-J05B3**; **T01-S03**

Original Publication Data by Authority

United States

Publication No. US 20030014397 A1 (Update 200334 B)
Publication Date: 20030116
Generating one or more XML documents from a relational database using XPath data model
Assignee: International Business Machines Corporation, Armonk, NY, US (IBMC)
Inventor: Chau, Hoang K., Sunnyvale, CA, US
Cheng, Isaac Kam-Chak, San Jose, CA, US
Cheng, Josephine Miu, San Jose, CA, US
Chiu, Suet Mui, San Jose, CA, US
Chow, Jyh-Herng, San Jose, CA, US
Pauser, Michael Leon, Morgan Hill, CA, US
Xu, Jian, San Jose, CA, US
Agent: Attention: George H. Gates, Gates & Cooper LLP, Howard Hughes Center, Suite 1050, 6701 Center Drive West, Los Angeles, CA, US
Language: EN (79 pages, 11 drawings)
Application: US 1999168659 P 19991202 (Related to Provisional)
US 2000725363 A 20001129 (Continuation of application)
US 200262127 A 20020131 (Local application)
Original IPC: G06F-7/00(A)
Current IPC: G06F-7/00(A)
Original US Class (main): 7073
Original Abstract: A technique is provided for creating metadata for fast search of XML documents stored as

column data. Data is stored in a data store connected to a computer. A main table is created having a column for storing a document, wherein the document has one or more elements or attributes. One or more side tables are created, wherein each side table stores one or more elements or attributes. Then, the side tables are used to locate data in the main table with scalable indexing mechanisms to facilitate search.A technique is provided for generating one or more XML documents from a single SQL query. Data stored on a data storage device that is connected to a computer is transformed. A query that selects data in the data storage device is received. The selected data is retrieved into a work space. Then, one or more XML documents are generated to consist of the selected data.A technique is provided for generating one or more XML documents from a relational database using the XPath data model. Data stored on a data storage device that is connected to a computer is transformed. Initially, a document object model tree is generated using a document access definition, which defines the mapping between an XML tree structure and relational tables. The document object model tree is traversed to obtain information to retrieve relational data. The relational data is mapped to one or more XML documents.A technique is provided to store fragmented XML data into a relational database by decomposing XML documents with application specific mappings. Data stored on a data store that is connected to a computer is transformed. Initially, an XML document containing XML data is received. A document access definition that identifies one or more relational tables and columns is received. The XML data is mapped from the application DTD to the relational tables and columns using the document access definition based on the XPath data model.
Claim: What is claimed is:

1. 1. A method of locating data in a data store connected to a computer, the method comprising the steps of:
 - creating a main table having a column for storing a document, wherein the document has one or more elements or attributes;
 - creating one or more side tables, wherein each side table stores one or more elements or attributes; and
 - using the side tables to locate data in the main table.

65/7/4 (Item 4 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

SAME APPLICANT

0015312757 *Drawing available*
WPI Acc no: 2005-662961/200568
Related WPI Acc No: 2002-010186
XRPX Acc No: N2005-543014

Publishing method of relational data as extensible markup language in computer system, involves generating extensible markup language output based on structured query language query results and tagging instructions

Patent Assignee: INT BUSINESS MACHINES CORP (IBMC)
Inventor: CAREY M J; IYER N; SHANMUGASUNDARAM J; SHEKITA E J

Patent Family (1 patents, 1 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
US 6947945	B1	20050920	US 2000531802	A	20000321	200568	B

Priority Applications (no., kind, date): US 2000531802 A 20000321

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
US 6947945	B1	EN	15	9		

Alerting Abstract US B1
NOVELTY - The **relational database** tables are **mapped** to virtual **extensible markup language (XML) documents**. The **XML** queries are issued, parsed and converted into language neutral intermediate representation. The representation is rewritten in equivalent form. The equivalent form is translated into structured query language (SQL) **query** and tagging instructions. The **XML** output is **generated** based on SQL **query** results and instructions.
DESCRIPTION - INDEPENDENT CLAIMS are also included for the following:

1. relational **data** publishing system; and
2. **computer program** product comprising relational **data** publishing **program**.

USE - For publishing relational **data** as **extensible markup language**, hyper **text** markup language and standard generalized markup language in client such as personal computer, workstation, minicomputer and mainframe computer **connected** to server through network such as local area network, internet and system network architecture network of computer system.
ADVANTAGE - The relational **data** can be published as **XML**, efficiently.
DESCRIPTION OF DRAWINGS - The figure shows a block diagram of Xperanto system.

Title Terms /Index Terms/Additional Words: PUBLICATION; METHOD; RELATED; **DATA**; EXTEND;

LANGUAGE; COMPUTER; SYSTEM; GENERATE; OUTPUT; BASED; STRUCTURE; QUERY; RESULT; TAG; INSTRUCTION

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-017/30			Main		"Version 7"

US Classification, Issued: 707102000, 715513000

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): **T01-J14**; **T01-N03A2**; **T01-N03B4**; **T01-S03**

Original Publication Data by Authority

United States

Publication No. US 6947945 B1 (Update 200568 B)
Publication Date: 20050920
Using an XML query language to publish relational data as XML
Assignee: International Business Machines Corporation, Armonk, NY, US (IBMC)
Carey, Michael James, San Jose, CA, US Residence: US
Shanmugasundaram, Jayavel, San Jose, CA, US Residence: US
Shekita, Eugene Jon, San Jose, CA, US Residence: US
Iyer, Narayana, San Jose, CA, US Residence: US
Inventor: Carey, Michael James, San Jose, CA, US Residence: US
Shanmugasundaram, Jayavel, San Jose, CA, US Residence: US
Shekita, Eugene Jon, San Jose, CA, US Residence: US
Iyer, Narayana, San Jose, CA, US Residence: US
Agent: McSwain, Marc D.
Language: EN (15 pages, 9 drawings)
Application: US 2000531802 A 20000321 (Local application)
Original IPC: G06F-17/30(A)
Current IPC: G06F-17/30(A)
Original US Class (main): 707102
Original US Class (secondary): 715513
Original Abstract: A method for publishing relational data as XML by translating XML queries into queries against an relational database. Conversion of the relational database into an XML database is not required. Each relational table is mapped to a virtual XML document, and XML queries are issued over these virtual documents. An XML query is parsed and transformed into a language-neutral intermediate representation, which is a sequence of operations describing how the output document is derived from the underlying relational tables. The intermediate representation is then translated into an SQL query over the underlying relational tables and into instructions for a tagger. The SQL query is executed, and the SQL query results are then fed into the tagger, which follows tagger

instructions to generate the marked up output.
Claim:

1. 1. A method of publishing relational data as XML, comprising the method steps of:
 - mapping a number of relational database tables to a number of virtual XML documents;
 - issuing XML queries over said virtual XML documents;
 - parsing said XML queries;
 - transforming said XML queries into a language-neutral intermediate representation;
 - rewriting said language-neutral intermediate representation into an equivalent form easily translated into an SQL query;
 - translating said equivalent form into an SQL query over said relational database tables and into tagging instructions passed to a tagger;
 - executing said SQL query to produce SQL query results passed to said tagger; and
 - generating XML output using said SQL query results and said tagging instructions.

65/7/3 (Item 3 from file: 350) **Links**
Derwent WPIX
(c) 2006 The Thomson Corporation. All rights reserved.

SAME APPLICANT

0015616640 *Drawing available*
WPI Acc no: 2006-180813/200619
XRPX Acc No: N2006-155912

Content data formatting method for client device, involves transcoding generic content data using client device characteristic information associated with request, to produce transcoded content data

Patent Assignee: INT BUSINESS MACHINES CORP (IBMC)
Inventor: FLOYD R A; HIND J R; PETERS M L; TOPOL B B

Patent Family (1 patents, 1 countries)

Patent Number	Kind	Date	Application Number	Kind	Date	Update	Type
US 7003584	B1	20060221	US 2000611158	A	20000706	200619	B

Priority Applications (no., kind, date): US 2000611158 A 20000706

Patent Details

Patent Number	Kind	Lan	Pgs	Draw	Filing Notes	
US 7003584	B1	EN	13	5		

Alerting Abstract US B1
NOVELTY - The requested content **data** is identified from a request having client device characteristic information. The characteristic information is stored in a **data structure** and **transferred** to transcoding **servlet** by preamble **servlet** . A generic content **data** formed using the requested content **data**, is matched to the request. The generic content **data** is transcoded by transcoding **servlet** using the client device characteristic information.
DESCRIPTION - INDEPENDENT CLAIMS are also included for the following:

1. apparatus for formatting content **data**; and
2. **computer program** product storing instruction for formatting content **data**.

USE - For formatting content **data** for presentation on client device e.g. personal computer and desktop computer.
ADVANTAGE - Since the transcoding filter **servlet** is allowed to access request header information, the formatting of requested content **data** of client device is performed effectively.
DESCRIPTION OF DRAWINGS - The figure shows a block diagram of the **data** processing system.

Title Terms /Index Terms/Additional Words: CONTENT; **DATA**; FORMAT; METHOD; CLIENT; DEVICE; TRANSCODER; CHARACTERISTIC; INFORMATION; ASSOCIATE; REQUEST; PRODUCE

Class Codes

International Patent Classification

IPC	Class Level	Scope	Position	Status	Version Date
G06F-0015/16	A	I	F	B	20060101

US Classification, Issued: 709246000, 709217000

File Segment: EPI;
DWPI Class: T01
Manual Codes (EPI/S-X): **T01-N03B4**; **T01-S03**

Original Publication Data by Authority

United States

Publication No. US 7003584 B1 (Update 200619 B)
Publication Date: 20060221
Apparatus and method for accessing request header information using a transcoding filter servlet
Assignee: International Business Machines Corporation, Armonk, NY, US (IBMC)
Floyd, Richard A., Chapel Hill, NC, US Residence: US
Hind, John R., Raleigh, NC, US Residence: US
Peters, Marcia L., Raleigh, NC, US Residence: US
Topol, Brad B., Raleigh, NC, US Residence: US
Inventor: Floyd, Richard A., Chapel Hill, NC, US Residence: US
Hind, John R., Raleigh, NC, US Residence: US
Peters, Marcia L., Raleigh, NC, US Residence: US
Topol, Brad B., Raleigh, NC, US Residence: US
Agent: Yee, Duke W.
Herndon, Jerry W.
Tkacs, Stephen R.
Language: EN (13 pages, 5 drawings)
Application: US 2000611158 A 20000706 (Local application)
Original IPC: G06F-15/16(B,I,H,US,20060101,20060221,A,F)
Current IPC: G06F-15/16(B,I,H,US,20060101,20060221,A,F)
Original US Class (secondary): 709246 709217
Original Abstract: The present invention provides an apparatus and method for accessing request header information used to transcode servlet output. The apparatus and method of the present invention includes a preamble that stores request header information from a request sent by a client device. The request header information is then provided to the transcoder along with the generated content data. The transcoder then performs appropriate transcoding on the generated content data based on the request header information supplied by the preamble. The transcoded content data is then sent to the client device. In this way, the client device is able to obtain content from a much larger set of content sources than with conventional systems.
Claim: What is claimed is:

1. 1. A method of formatting content data for presentation on a client device, comprising:
 - receiving a request for content data, the request having client device characteristic information and

identifying requested content data;

- storing, by a preamble servlet, the client device characteristic information in a data structure associated with the request;
- passing, by the preamble servlet, the client device characteristic information to a transcoding servlet;
- generating the requested content data to form generic content data;
- matching the generic content data to the request; and
- transcoding, by the transcoding servlet, said generic content data using said client device characteristic information in the data structure associated with the request to produce transcoded content data.